Bioinformatics Integration Support Contract (BISC), Phase II

SEQUENCE FEATURE VARIANT TYPE (SFVT) ANALYSIS USER GUIDE

Version 2.0

Period Of Performance: September 30, 2004—September 29, 2010
Developed Under Contract Number: HHSN266200400076C
ADB Contract Number: N01-AI-40076
Delivered: August 7, 2009

Project Sponsor:
National Institutes of Health (NIH)
National Institute of Allergy and Infectious Diseases (NIAID)
Division of Allergy, Immunology, and Transplantation (DAIT)

NORTHROP GRUMMAN
Information Technology
Prepared by:
Federal Enterprise Solutions
Health Solutions
2101 Gaither Rd, Suite 600
Rockville, Maryland 20850
(301) 527-6600
Fax: (301) 527-6401
jeff.wiser@ngB.com

Contents

Appendixes

SOP for HLA Quality Control Pipeline Version History

Version	Date	Description
1.0	04/08/09	SFVT User Guide Initial Version
2.0	08/07/009	Updated for ImmPort Release 2.5.2

1.0 INTRODUCTION

This document is the user guide for Sequence Feature Variant Type (SFVT) vector file generation and HLA Quality Control (HLA QC) pipeline using the tool PyPop. Section 2 describes the use of the ImmPort's MHC SFVT analysis and the HLA QC pipeline. Section 3 provides the details on vector file generation process. Section 4 provides details on the HLA QC pipeline.

The sequence feature variant type approach is a novel method to annotate HLA allele types and for the analysis of HLA typing data. In this approach, for each classical HLA locus, several sequence features are defined based on structural (e.g. beta-sheet 1, alpha-helix 1, etc), functional (e.g. peptide antigen binding, CD4 receptor binding, etc) and sequence altering (insertions, deletions, single amino acid variations) information. These sequence features are defined by amino acid positions with reference to the mature protein sequence of a reference allele for a locus. The sequence features can be overlapping and continuous or discontinuous in the linear sequence. The extent of sequence variation among HLA alleles was then assessed for each HLA sequence feature to define all variant types (VT) found in the human population. Thus, an HLA allele can be annotated with the sequence feature variant types (SFVTs) of the different sequence features defined for the locus. The following are the benefits of this approach:

i. The intricate relationships among the alleles of the highly polymorphic HLA genes can be studied through the variety of sequence feature variant types shared by alleles
ii. From a clinical study, the 4 digit HLA allele types for individuals can be annotated with the sequence feature variant types. This allows calculating the frequencies of the SFVTs and subsequently, in performing an association analysis to determine, with higher statistical power, the molecular determinants associated with a disease or any other clinical phenotypes. Thus, by this approach, parts of the HLA allele molecule associated with the disease can be determined and also elucidate the alleles that share a particular significantly associated motif sequence.

The MHC SFVT Analysis Tool takes HLA typing data and converts the identified HLA alleles into their component SFVTs. For example, for HLA-DRB1*0701, amino acid motif sequence defined at the "beta-strand 2_peptide antigen binding pocket 4" sequence feature (Hsa_HLA-DRB1_SF155 defined by positions 26, 28) of the protein is 'FE' (Hsa_HLA-DRB1_SF155_VT2 i.e. second variant type of all variant types found at this sequence feature for the DRB1 locus). Thus, the DRB1*0701 allele is annotated with "Hsa_HLA-DRB1_SF155_VT2" for the SF155. Similarly, the variant types for rest of the 181 sequence features of HLA-DRB1 locus are determined and annotated for HLA-DRB1*0701.

2.0 SFVT VECTOR GENERATION AND HLA QC USING PYPOP

Through this tool one can take HLA Typing data and generate sequence feature variant types for each allele of the classical loci in the dataset or run the HLA QC pipeline. The set of

sequence feature variant types for an allele is referred to as the SFVT vector. There are three main steps to follow to effectively use this tool. They are illustrated below in figure 1.

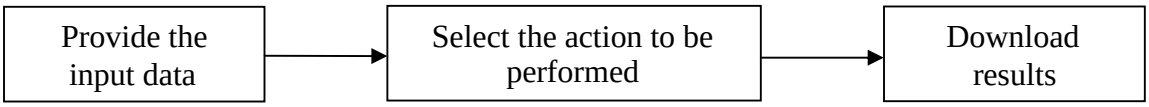


Figure 1 Schematic of the steps in using the SFVT vector generation tool

2.1 INPUT DATA SOURCES

The input HLA typing data can be provided to the ImmPort system through various data sources. Currently, the tool supports provision of the data through three types of actions:

a) Uploading HLA typing file
b) Creating HLA typing file from subjects' data stored in ImmPort
c) Selecting HLA typing file stored in ImmPort

2.1.1 Uploading HLA typing file

This option allows the user to upload the HLA typing results from the local machine. This file can be retrieved later for re-use by following the workflow described in 2.1.3. This Section describes the steps involved in providing the uploaded HLA typing file for further processing.

Select the "Upload HLA Typing Data File" option as shown in figure 2A and proceed to the next step. Select the project (see figure 2B) to which the uploaded input file and the generated results will be stored.

It is necessary to make sure the typing results uploaded are presented in a format acceptable by the tool. The user can either download the HLA typing results template "HLA_Typing.xls" from the upload page (see figure 2B) and fill it with the HLA typing results or edit one's input file to reflect the format shown in the "HLA_Typing.xls" (see Appendix A. for the supported formats). The uploaded file can be given a custom file name and be uploaded either as an Excel 2003 file or a tab-delimited text file.

Figure 2A Select the data source

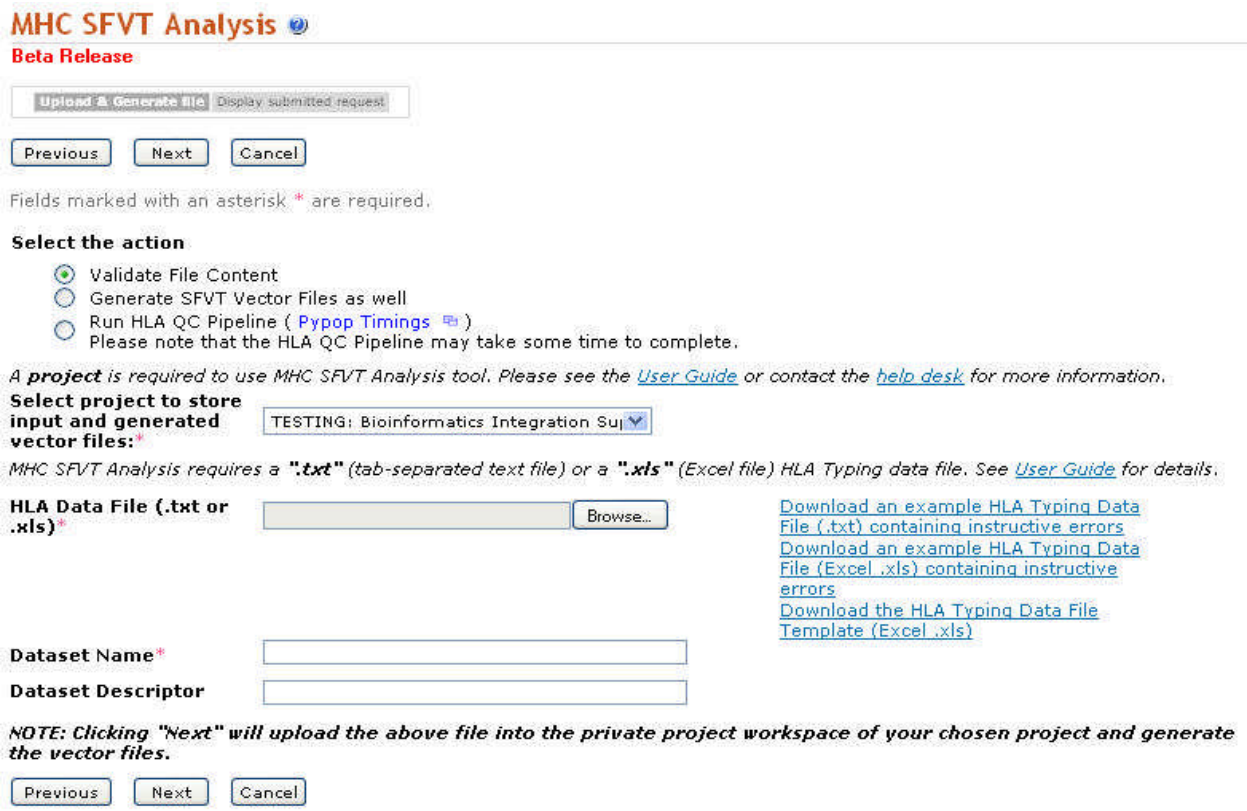


Figure 2B HLA typing results upload

As shown in figure 2B it is optional to provide description of the uploaded dataset and the submitted task. This helps in identifying the results from the analysis history as shown in figure 6B.

2.1.2 Subject's HLA typing data in ImmPort

This option allows one to create the input HLA typing file for the analysis tool from the HLA typing data of the subjects in ImmPort. The final created HLA typing file can be retrieved later for re-use by following the workflow described in Section 2.1.3. This Section describes the steps involved in creating the HLA typing file for input to the analysis tool.

Select the "Create HLA Typing Data File" option from the "select data source" options as shown in figure A and proceed to the next step.

a) Subjects can be selected from different projects or from pre-defined lists of subjects. (Subjects list can be created by first searching for subjects using advanced research data search resource of ImmPort and then saving the selected subjects from each search to a list.) To select the subjects whose typing data is to be analyzed, select the projects or the subject lists that contain the subjects of interest. Then, click "Load from selected project(s)" or "Load from selected list(s)" depending on the subject data source as shown in figure 3A.
b) From the list of subject records select the subjects whose typing data is to be analyzed and click the "Submit" button.
c) Once the subjects are selected the loci typed for the selected subjects are shown in figure 3C. Select the HLA locus whose typing data is to be analyzed.

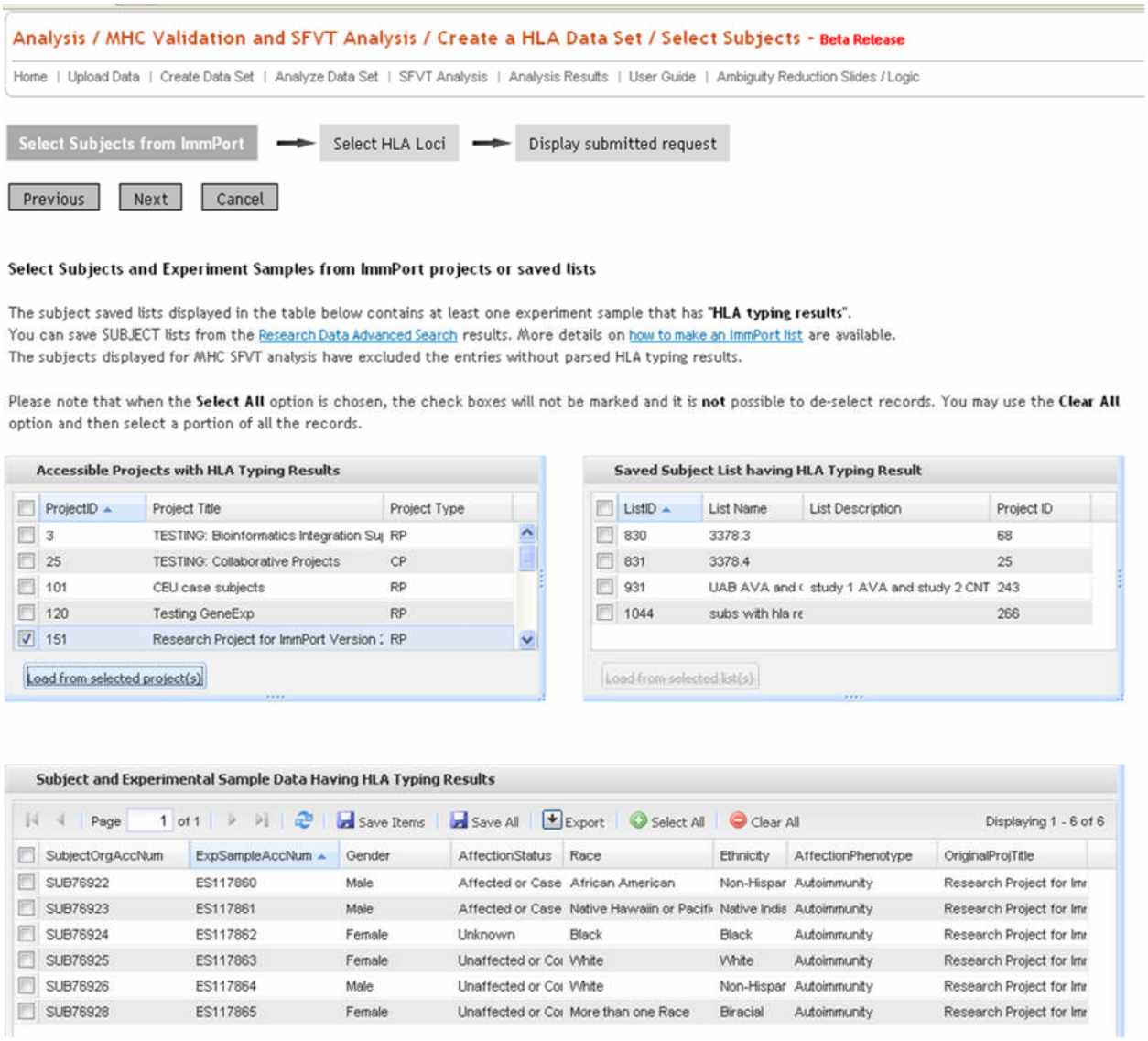


Figure 3A Load the subjects from Projects or Subject Lists

Figure 3C Select the locus whose HLA typing data needs to be analyzed

2.1.3 HLA typing files stored in ImmPort

This option allows one to select an already uploaded or created HLA typing file that is ready for analysis by this tool. The steps involved are: first, select the "Use HLA Typing Data File in ImmPort" option from the "select data source" options shown in figure 2A. Then, select the HLA typing file from the table of existing files in ImmPort as shown in figure 4.

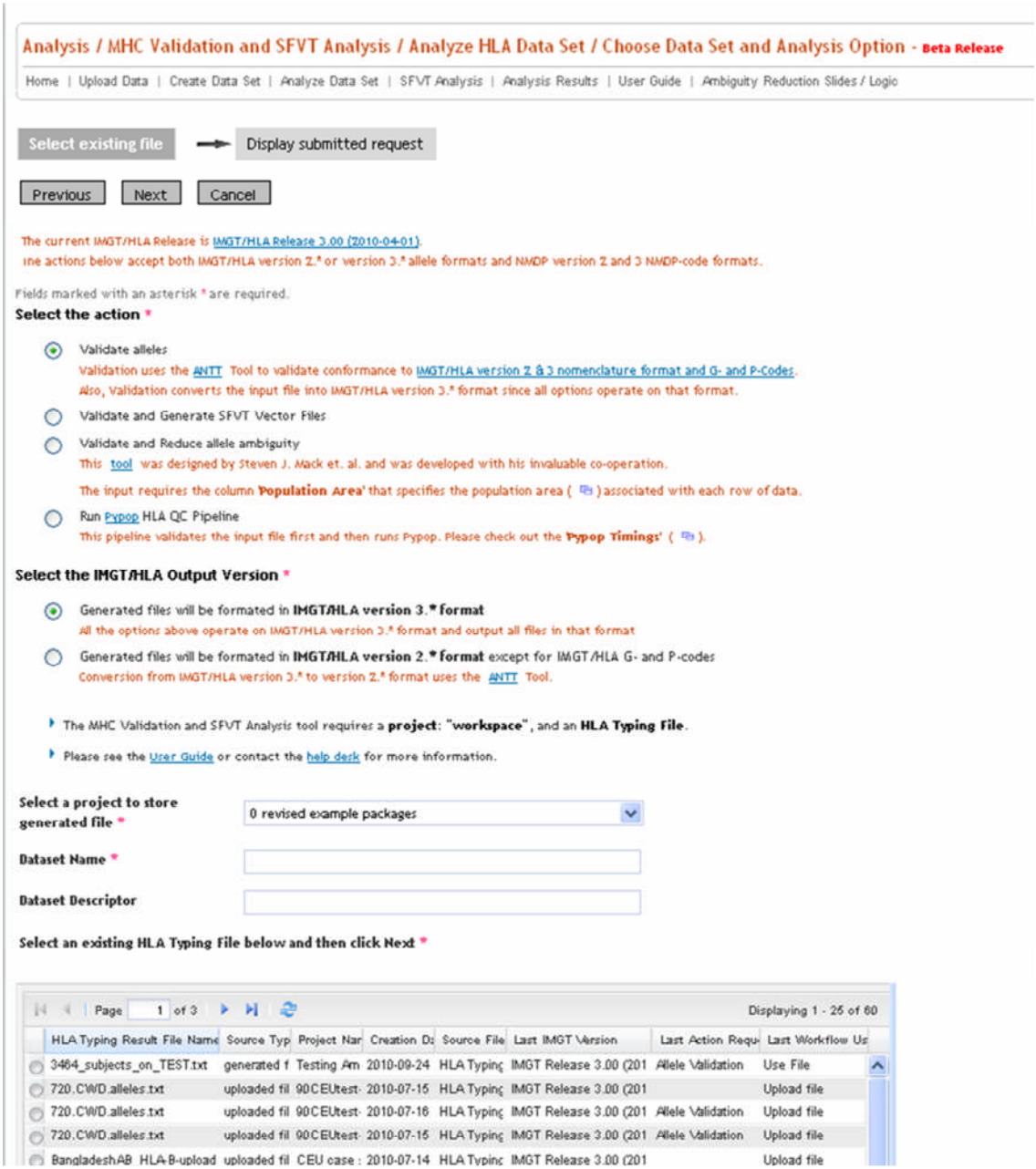


Figure 4 Select the HLA typing file ready for analysis from the existing files in ImmPort

2.2 ANALYSIS OPTIONS AVAILABLE

Once the input HLA typing file is made available for the tool through any of the various sources as described in Section 2.1, the next step in the workflow is to select the type of analysis to be performed. Irrespective of the source of input HLA typing file there are three options for the analysis to be performed as shown in figure 5 which are described below in this Section.

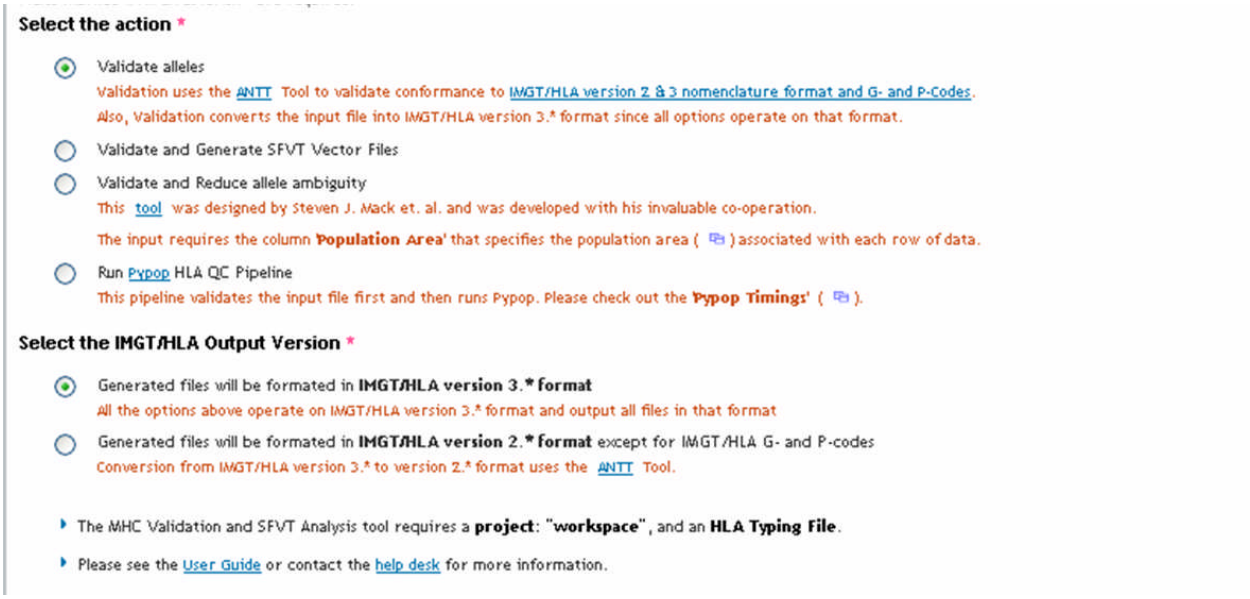


Figure 5 Select the analysis to be performed

Once the type of analysis action is selected, select the project to where the analysis results need to be stored. These results can then be retrieved later.

i. **Validate File content:**
   If the HLA alleles in the input HLA typing file need only to be validated then select this action. The validation process is described in the Sections 3.1 and 3.2.

ii. **Generate SFVT Vector Files as well:**
   In addition to the validation of the alleles in the input HLA typing file, one can also generate the SFVT vectors for the alleles, this analysis option is selected. The process and the output files that are generated are described in Sections 3.3 and 3.4.

iii. **Validate alleles and reduce allele ambiguity:**
   The allele ambiguity reduction based upon work from Steven J. Mack, Ph.D..

iv. **Run HLA QC Pipeline:**
   ImmPort currently supports the Quality Control (QC) of HLA typing data using the PyPop software developed by [Lancaster, R.M. et al] *(A. K. Lancaster, R. M. Single, O. D. Solberg, M. P. Nelson and G. Thomson 2007 "PyPop update - a software pipeline for large-scale multilocus population genomics" Tissue Antigens 69 (s1), 192-197.)*
   The files generated by the QC analysis are described in Section 4 and more information about the tool can also obtained by referring to the PyPop tool's user

guide available at the following site: http://www.pypop.org/.
This analysis option does not generate the SFVT vector files but generates comprehensive quality control analysis results.

2.3 ANALYSIS RESULTS

Upon submission of the task, a task ID is assigned (as shown in figure 6A) to the request to generate the SFVT vector file. This task ID or the dataset name can be used to monitor the status of the task in the analysis history as shown in figure 6B.

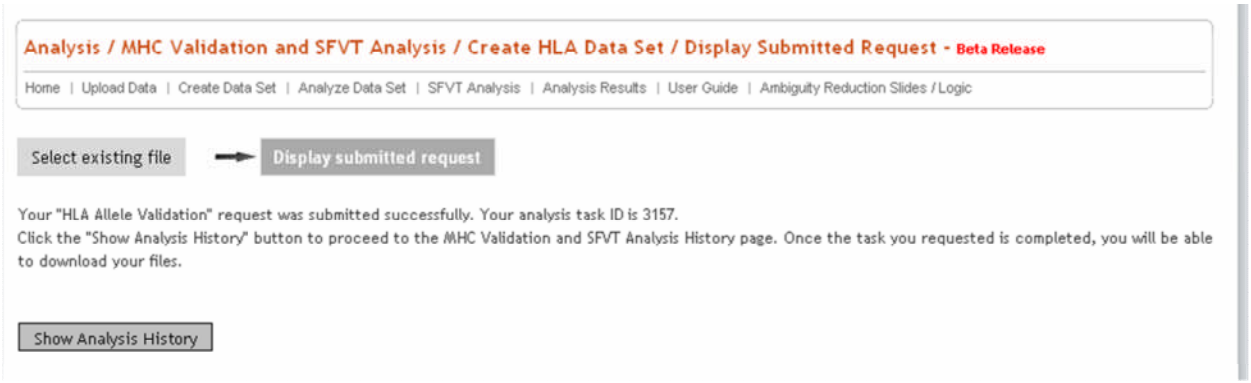


Figure 6A Confirmation of Task submission

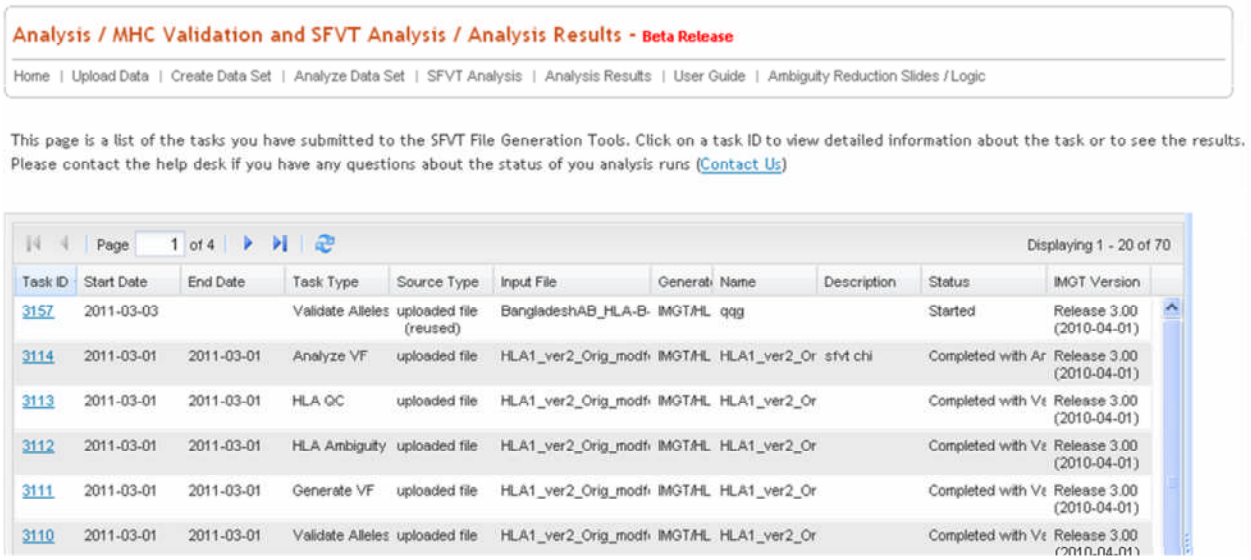
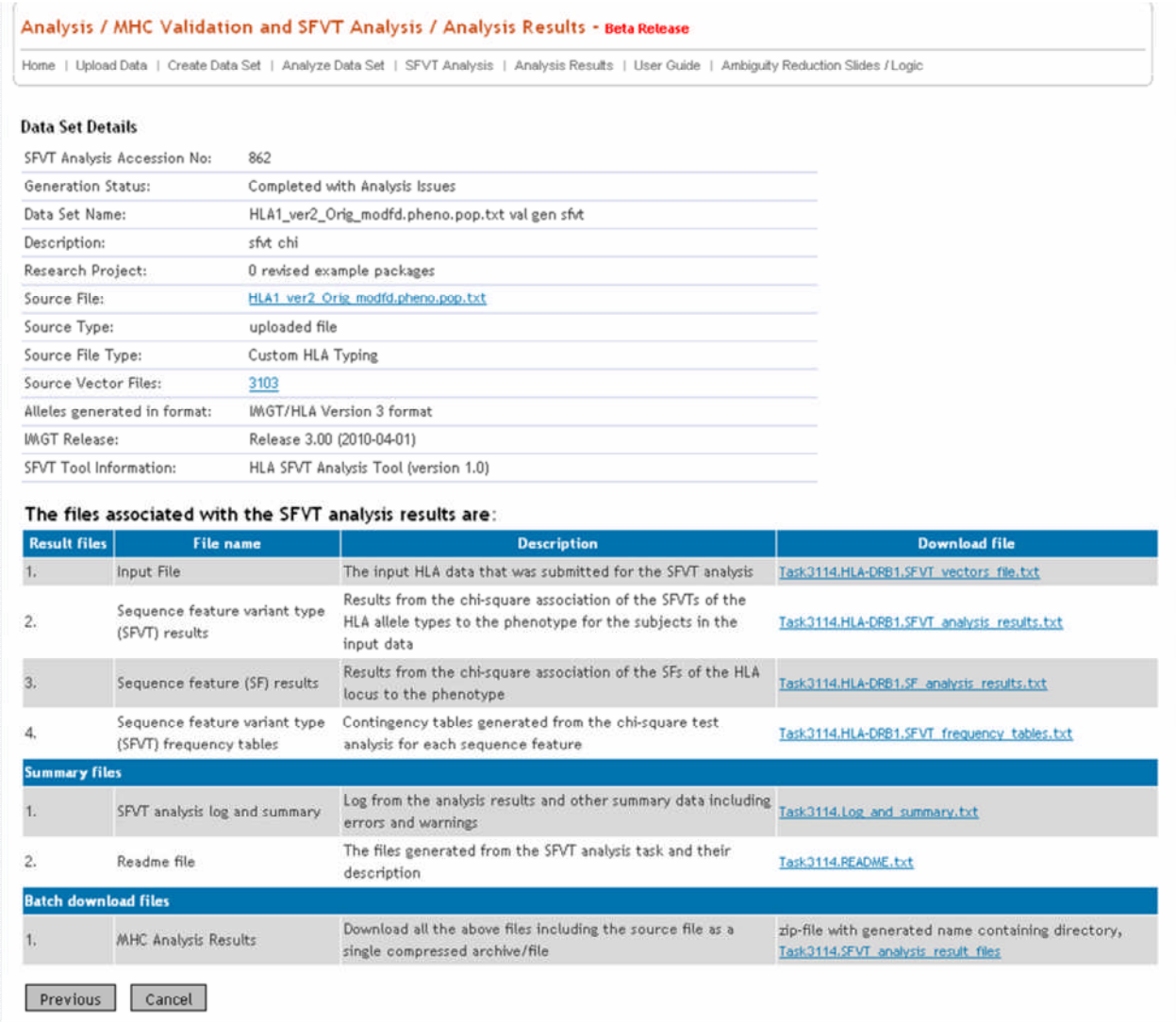


Figure 6B Analysis history table

To download the results select the task of interest and click the task ID to proceed to the download page shown in figure 7. Depending on the analysis action selected different set of files are available for download as shown in figures 7. There are files generated that give reports on

any errors encountered in the analysis process. The details of the MHC SFVT Analysis process, the content of the files and the errors are provided in Sections 3.0 and 4.0 and in the Appendix B.



Analysis / MHC Validation and SFVT Analysis / Analysis Results - Beta Release

Home | Upload Data | Create Data Set | Analyze Data Set | SFVT Analysis | Analysis Results | User Guide | Ambiguity Reduction Slides / Logic

Data Set Details

SFVT Analysis Accession No:	862
Generation Status:	Completed with Analysis Issues
Data Set Name:	HLA1_ver2_Orig_modfd.pheno.pop.txt val gen sfvt
Description:	sfvt chi
Research Project:	0 revised example packages
Source File:	HLA1_ver2_Orig_modfd.pheno.pop.txt
Source Type:	uploaded file
Source File Type:	Custom HLA Typing
Source Vector Files:	3103
Alleles generated in format:	IMGT/HLA Version 3 format
IMGT Release:	Release 3.00 (2010-04-01)
SFVT Tool Information:	HLA SFVT Analysis Tool (version 1.0)

The files associated with the SFVT analysis results are:

Result files	File name	Description	Download file
1.	Input File	The input HLA data that was submitted for the SFVT analysis	Task3114.HLA-DRB1.SFVT_vectors_file.txt
2.	Sequence feature variant type (SFVT) results	Results from the chi-square association of the SFVTs of the HLA allele types to the phenotype for the subjects in the input data	Task3114.HLA-DRB1.SFVT_analysis_results.txt
3.	Sequence feature (SF) results	Results from the chi-square association of the SFs of the HLA locus to the phenotype	Task3114.HLA-DRB1.SF_analysis_results.txt
4.	Sequence feature variant type (SFVT) frequency tables	Contingency tables generated from the chi-square test analysis for each sequence feature	Task3114.HLA-DRB1.SFVT_frequency_tables.txt
Summary files			
1.	SFVT analysis log and summary	Log from the analysis results and other summary data including errors and warnings	Task3114.Log_and_summary.txt
2.	Readme file	The files generated from the SFVT analysis task and their description	Task3114.README.txt
Batch download files			
1.	MHC Analysis Results	Download all the above files including the source file as a single compressed archive/file	zip-file with generated name containing directory, Task3114.SFVT_analysis_result_files

Previous Cancel

Figure 7 Downloadable files generated for the MHC SFVT analysis action

3.0 SFVT VECTOR FILE GENERATION PROCESS

This Section describes sequence feature variant type (SFVT) vector file generation process. Individual files, one per locus, containing SFVT vectors are generated for each set of HLA allele typing data. HLA Typing data are provided in the file formats as specified in Appendix A.

Before an HLA typing data can be transformed into the SFVT vector, it is necessary to validate that the typed HLA alleles are in compliance with the latest report of the *Nomenclature*

for Factors of the HLA System (http://hla.alleles.org/nomenclature/nomenc_reports.html). In addition to the SFVT vector files, following files are generated:

i. *Allele validation output* – This file is similar to the input HLA typing results file except that only the validated alleles are included and this will be the file used for generating the SFVT vector files.
ii. *Allele validation summary* – This is a report of the validation on the alleles in the input dataset. It contains information on the number of typed alleles that have errors and other useful information such as the number of common and rare alleles present for each HLA locus typed in the dataset
iii. *MHC SFVT summary* – A report of any errors in the generation of SFVT vectors are presented in this summary along with information on allele summary

The following Subsections describe the validation and vector file generation process including the files that are generated.

3.1 ALLELE VALIDATION—ALLELE NAME SYNTAX

The valid syntaxes of an HLA allele entry are described through Subsections '3.1.1 Valid Allele syntax' and '3.1.2 Syntax descriptors'. The syntax validation process is described in Subsection 3.1.3. Error messages for this process are explained in Appendix B.1 "Allele Validator Errors" and Appendix B.4 "Allele Errors".

3.1.1 Valid Allele syntax

The valid allele syntaxes are specified in Table 1, "Valid HLA Allele Syntaxes". In the table below, white space (spaces) can occur around any of the components. This white space is ignored in processing.

Table 1, Valid HLA Allele Syntaxes

Entry item	Syntax	Notes
<allele_names>	i) Full name - <hla_locus>*(3 or 4 or more digits name), e.g. "DRB1*0101", "DRB1*010101", "DRB1*101" ii) Only the digits – (3 or more digits), e.g. 0101, 010101, 101 iii) Allele names with suffixes - <allele name><name_suffix> e.g. "DRB1*1613N" or "1613N" iv) Multiple alleles - <allele_name><allele_separator><allele_name><allele_separator>…. E.g. "DRB3*0101/0202", "0101/0202", "DRB3*0101/0102/0203/…"	For 3 or 5 digits names '0' is assumed as the initial digit. If more than 5 odd number of digit names, it is assumed to be specified as is.
<nmdp_alleles>	<hla_locus>*<digit><digit><nmdp_code> e.g. DRB1*01TP	

Entry item	Syntax	Notes
<gcode_alleles>	i) Full name <hla_locus>*(4 or more digits name)<gcode_suffix> e.g. A*2601g ii) Only the digits - (4 or more digits name)<gcode_suffix> e.g. 2601g	
<serological_alleles>	i) Full name - <hla_locus>*<digit><digit> e.g. DRB1*01 ii) Only the digits – (1 or 2 digits), e.g. 01, 1	

3.1.2 Syntax descriptors

The accepted values for the allele separator, digits, name_suffix, and hla_locus are defined in Table 2, "Syntax descriptors". In the specification below, alphabetic characters are shown as upper-case, but all comparisons are performed in a case-sensitive manner. Also, for **<missing_allele>** white space (spaces) is ignored in processing.

Table 2, Syntax descriptors

Descriptor	Accepted values	Notes
<allele_separator>	':' or '/' or ',' or ' '	For a given entry, only one type of separator can be used to separate alleles
<digit>	'0' or '1' or '2' or '3' or '4' or '5' or '6' or '7' or '8' or '9'	
<gcode_suffix>	'g'	Standard g-code specifier suffix
<hla_locus>	'HLA-A' or 'A' or 'HLA-B' or 'B' or 'Cw' or ...	Standard Anthony Nolan HLA locus names
<missing_allele>	• HLA Typing Result: '' or '-' or '"-"' or 'XXXX' • PyPop input file: '****'	
<name_suffix>	'C' or'N' or 'L' or 'S' or 'A' or 'Q'	Standard Anthony Nolan HLA allele nomenclature suffix
<nmdp_code>	'AB' or 'AC' or ...	Standard NMDP code names; these codes differ from **<name_suffix>** values so are distinguishable

3.1.3 Syntactic Validation

Syntactic validation determines if there is a valid entry using Table 1 and 2 above. This validation does not determine whether or not the alleles represented by an entry are valid Anthony Nolan allele names (See Section 3.2 for description on that validation step). If the syntax validation fails, error messages are generated and no further processing of the entry is performed. Also, during syntactic validation processing, certain data transformations occur. For example, the alleles represented in NMDP codes will be transformed to its representative alleles.

3.2 Allele Validation—Validating Allele Names

This Section describes the process of validating the allele names for conformance to Anthony Nolan allele names. Once an allele entry has been syntactically validated, then the set of Anthony Nolan allele names represented by this entry is determined. Any errors are reported (See Appendix B.1 "Allele Validator Errors" and Appendix B.5 "Allele Errors"). For **<nmdp_alleles>** and **<gcode_alleles>** the alleles are looked up against the corresponding

standard (see Section 3.2.1 and 3.2.2). In the case of **<nmdp_alleles>**, a transformation into a set of alleles represented by the code is performed. For an odd digits name entry, a two step process is followed: a zero digit ('0') is prefixed to the digits and the allele is checked against the current alleles, and failing that the allele is checked in the Anthony Nolan changed names list.

3.2.1 NMDP Code Transformation

The NMDP codes are translated using the lookup table provided at the NMDP web-site http://bioinformatics.nmdp.org/HLA/allele_code_lists.html. The process to translate a code depends on whether the value of the code is a set of 2-digit or 4-digit values. For example, if the NMDP code is B*58VE, the lookup for 'VE' will return the value '01/11'. The alleles for the code will be B*5801 and B*5811.

Another example is B*15BKVK. The lookup for the code, 'BKVK', returns the value '1501/1501N/9502/9504'. The alleles for this code are B*1501, B*1501N, B*9502, and B*9504.

If the NMDP code is not known, then an error is reported and the entry is not processed further. Otherwise if the NMDP code is defined and consistent for the locus, then it is replaced in the validated file by its set of alleles.

3.2.2 G-Code Lookup

The determination of alleles for a g-code is a grouping code determined by a lookup table derived from the paper [Cano et al: 2007] ***("Common and Well-Documented HLA Alleles", Human Immunology 68, 392-417 2007)***. For example, if the g-code A*020101g is provided, then the alleles A*0201, A*02010101, A*0209, A*0243N, A*0266, A*0275, A*0283N, and A*0289 are returned. If the g-code is not known, then an error is reported and the entry is not processed further. The gcode is left as-is in the validated file.

3.2.3 Special Names Replacement

Special names fall into two categories:

- 'Suggested Name' as defined in the paper [Cano:etal:2007]
- 'Code in Table' as defined in the Anthony Nolan ambiguous typing data

In the first case, the 'Suggested Name' is replaced by its corresponding g-code as defined in Section 3.2.2, and in the second case the 'Code in Table' is replaced by the list of alleles defined in the Anthony Nolan ambiguous typing data. This ambiguity typing data is available from the ANT web-site for the current version of the HLA allele data: http://www.ebi.aB.uk/imgt/hla/ambig.html

3.2.4 Allele Name Lookup

Once the allele names are successfully determined, they are then checked to see if they exist in the current release of the Anthony Nolan HLA dataset. There are several cases specified

in priority checking order in Table 3, "Allele Name Validation Cases" below. Only the first case will represent a clean match. The other cases will require further processing. The changed allele name and deleted allele name lists specified in the table below for a given Anthony Nolan release are acquired from the Anthony Nolan web site: **ftp.ebi.aB.uk/pub/databases/imgt/mhc/hla**

Table 3, Allele Name Validation Cases

Case	Description
Exact match	The allele name represents exactly one current allele
Multiple matches	More than one current allele is matched. This is not an error, but will be reported for further processing.
Delete match	The allele name matches with a deleted allele in ANT. If the deleted allele references a replacement allele, then this is not an error, but it will be reported along with the replacement name for further processing. If the deleted allele does not reference a replacement allele, then an error is generated.
Change match	The allele name matches with one of the changed names in ANT. This usually occurs for the allele names with odd number of digits of either five (5) or seven (7). This is not an error, but it will be reported along with the replacement name for further processing
missing match	None of the above categories apply. An error is generated.

3.2.5 Examples of allele entries

The table 5, "Validation examples for allele entries" displays examples of valid and invalid entries that may occur for the HLA-A locus and the results from validation that will be performed. An error is indicated by an empty validated result in the table. Also, Table 4, "Valid allele entries for HLA-A", illustrates the valid entries for the HLA-A locus.

Table 4 Valid allele entries for HLA-A

Allele entry	Alleles Represented
A*0110	A*0110
0110	A*0110
0110/0106	A*0110 and A*0106
A*0110/0106	A*0110 and A*0106
A*2312/14/15	A*2312, A*2314, and A*2315
110	A*0110
110/106	A*0110 and A*0106
110/06	A*0110 and A*0106
A*02AMJM	A*0201, A*0209, A*0243N, and A*0266
A*010101g	A*01010101 and A*0104N
A*01XX	All A* alleles with serological category '01'

Table 5 Validation examples for HLA-A allele entries

Original Entry	Validated Result
010101g/A*0101	
0110	0110
0110/0106	0110/0106
0110/106	
02	02
0209/43N	0209/0243N
0294N	0294N
02AMJM/A*0101	
03	03
03/02	
0300	03
0300/02	
1010.0	
1010102N	01010102N
110	0110
110/0106	0110/0106
110/06	0110/0106
110/106	
2	02
2202	
2402101	24020101
294N	0294N
3	03
3/02	
300	03
300/02	
3013	3013
5101/17/21	
68011/0101	680101/0101
68011/2402101	680101/24020101
A*0101.1	0101
A*0101.1N	
A*0101/0200	
A*0101/02AMJM	
A*0101/A*0101011g	
A*0101/A*010101g	
A*0101/A*02	
A*0101/A*0200	
A*0101/A*0212AMJM	
A*0101/A*102	

Original Entry	Validated Result
A*0101/A*3	
A*0101/A*B03	
A*0101011g	
A*0101011g/A*0101	
A*010101g	010101g
A*010101g/A*0101	
A*010102g	
A*0110	0110
A*0110/0106	0110/0106
A*02	02
A*0200N	
A*020101g	020101g
A*02011	020101
A*020120	020118
A*0212AMJM	
A*0212AMJM/A*0101	
A*021AMJM	
A*02202	022002
A*0294N	0294N
A*02AMJM	0201/0209/0243N/0266
A*02AMJM/A*0101	
A*02BRHJ	0201/0209/0243N/0266/0275/0283N/0289
A*02N	
A*03	03
A*03/02	
A*0300	03
A*0300/02	
A*03013	030103
A*0312345678	
A*03BRHJ	
A*03VS	0301/0320
A*101/A*102	
A*10101g	
A*10102g	
A*110	
A*2	
A*200N	
A*20102	020102
A*2202	
A*2312/14/12/14	2312/2314
A*2312/14/15	2312/2314/2315
A*2401	

Original Entry	Validated Result
A*2402101/02L	24020101/24020102L
A*24022	240202
A*2901102N	29010102N
A*29011N	
A*294N	
A*2N	
A*3	
A*3	
A*3/02	
A*3/A*0101	
A*300	
A*300/02	
A*3013	3013
A*3021	301102
A*312345678	
A*B03	
A*B03/A*0101	
A*11XX	11
A*11xx	11
11XX	11
11xx	11
A*11XX/A*2301	
11XX/2301	
A*2301/A*11XX	
2301/11XX	
A*11xx/A*2301	
11xx/2301	
A*2301/A*11xx	
2301/11xx	
A*0102	0102
0102	0102
102	0102
0102/0106/0103/0110	0102/0106/0103/0110
0201, 0209, 0243N, 0266	0201/0209/0243N/0266
A*0102/A*0103/A*2612	0102/0103/2612
A*0104N/A*02010101	0104N/02010101
01010102N/010104/0117/020107	01010102N/010104/0117/020107
010105	010105
A*020170	
A*260101/2624/2626	2601g
A*02G1	020101g
A*7401/7402	7401g

Original Entry	Validated Result
A*1101/1121N	110101g

3.3 SFVT VECTOR GENERATION

This Section describes the rules and the process for the vector file generation. In the generated file, an element of the vector can be one of the following categories of values as described in Table 6, "Values of vector elements" below:

Table 6, Values of vector elements

Vector element Value	Description
Defined value	Exactly one variant type for the feature, for example "Hsa_HLA-DRB1_SF1_VT10"
Type Unknown	The sequence feature variant type is unknown due to lack of sequence information. E.g. "Has_HLA-DRB1_SF1_Type Unknown"
"Null"	Singleton allele is a NULL allele
"No Data"	Singleton allele in the entry for an individual is a serological code (2-digt code)
"Ambiguous Type"	The individual is typed with more than one allele and hence the variant type is ambiguous.

Thus, each validated allele entry is checked to see if they are typed at serological level or if it is a null allele or has multiple allele types. In either of the cases the following steps are taken:

1. Serological data (2-digit allele) always generates 'No Data' for the variant type of all the sequence features of that locus
2. A NULL allele (for example, A*01010102N) always generates 'Null' for the variant type of all the sequence features of that locus
3. For allele entries with multiple allele types, the "Ambiguous Type" is assigned for the variant type of all sequence features of that locus except for those sequence features for which the multiple alleles share a similar sequence feature variant type.
4. An allele that is neither NULL nor serological will generate a variant type (defined value) as defined by its protein (4-digit code). For example, A*010101 will generate the sequence feature variant type for each feature assigned to the protein A*0101. Some variant type assignments for some alleles are designated as 'Type Unknown' if the sequence information is unknown at any of the amino acid positions that define the sequence feature.

3.4 VECTOR FILE GENERATION OUTPUTS

The vector file generation pipeline creates the following files which are described in the Subsections below:

- **validated file**
- **validation summary**
- **vector files**
- **generation summary**

3.4.1 Validated File

The uploaded input file containing the allele types for every individual in the dataset is first validated as described in Section 3.2. A file is generated as a result of the validation and only the validated entries are considered for SFVT vector generation. This file has the same format as the file input to the MHC SFVT Analysis process and is identified by the following name syntax: "**<Task_ID>**.allele.validation_output.txt" where <Task_ID> is the ID assigned to the task running the MHC SFVT analysis process. The task IDs along with other related information are listed in the analysis history page. However, the allele content has been modified to represent the validated alleles as shown for few examples in Table 5. An example of a validated file is illustrated in Table 7, "Example of a Validated File" below.

Table 7, Example of a Validated File

HLA Typing Results												…
Please do not delete or edit this column												…
Column Name	Experiment Sample User-Defined ID*	HLA-A Allele 1	HLA-A Allele 2	HLA-B Allele 1	HLA-B Allele 2	HLA-C Allele 1	HLA-C Allele 2	HLA-DPA1 Allele 1	HLA-DPA1 Allele 2	HLA-DPB1 Allele 1	HLA-DPB1 Allele 2	…
	AB1234	0201	0101	4402	0801	0501	0701			0401	0401	…
	AB2345	0101		0801	4402	0701	0501			0201	0201	…
	AB3456	0101	2402/2403	0801	3502	0701	0401			0402	0401	…
	AB4567	0201	0202	4901	0801	0701	0701			0301	0401	…
	AB5678	0301/0308	0201/0234	2702	4402/4419N	020202	0501			0402	0402	….

Note that, the empty cells are replaced with an empty column, and allele names are presented numerically (no prefix) in the standard IMGT format. Changed and deleted names are replaced by their current allele names. NMDP codes are replaced by their IMGT allele equivalents. Finally, any cell that contained data that can not be validated successfully is replaced by an empty cell.

3.4.2 Validation Summary

The validation summary file, **Allele.validation_summary_Log.log**, contains the following informational tables occurring in the summary file in the order specified in Table 8, "Informational Tables in Validation Summary". These tables describe the results of the validation process.

Table 8, Informational Tables in Validation Summary

Table Name	Description
Version information	This tables contains the following information: i. Versions of the source data utilized in validating the input alleles ii. Version of the vector file generator tool iii. Start time of the task
Files and species information	This table provides the species information, the input file type and the list of generated files
Subject Statistics	This table provides the number of subjects present in the input data
Summary of Error Counts by Locus	This table provides the number of typed alleles, by locus, that were validated successfully (**No Er**ror) and unsuccessfully (**Erro**r).
List of Errors	This table only appears if there were errors in validating the allele data. This table lists all errors that occurred with the following column information: **Row Num**—data row number in the file **Row Id**—the 'subject ID' or the 'Experiment Sample User-Defined ID' **Col Name**—the column name in the input file identifying the locus and the allele **Allele entry Data**—the allele entry **Err Num**—the error number of the error as defined in Appendix B **Error Message**—a short error message
Validation Notes for entries	This table only appears if there were processing notes that occurred in validating the allele data. This table lists all notes that occurred with the following column information: **Row Num**—data row number in the file **Row Id**—the 'subject ID' or the 'Experiment Sample User-Defined ID' **Col Name**—the column name in the input file identifying the locus and the allele **Allele entry Data**—the allele entry **Note Message**—a note about the issue that occurred
Error and Note Details by entries	This table of information only occurs if there were errors or notes that occurred in the validation process. The error messages that can occur in this table are specified in detail in Appendix B
Category Errors	This table only occurs if there were any errors. It provides the counts of each error by error category. The error categories and error numbers in this table correspond to those listed in Appendix B.

Table Name	Description
Summary of Data Types By Locus	This table defines the count of the special data types that occurred per Locus. The data type include: **Changed Name**—a changed name as defined by IMGT **Deleted Name**—a deleted or replaced name as defined by IMGT **G-Code**—as defined in Section 2.2.2 **NMDP Code**—defined by in Section 2.2.1 **NULL Allele**—a NULL allele as defined by IMGT **Serological**—a 2-digit allele only specifying the allele's serological category **Zero Prefix**—a 3-digit allele that is missing its zero-prefix
Summary of Common and Well-Documented (CWD) Data by Locus	This table provides the number of CWD and rare alleles per Locus. *Common and Well-Documented (CWD)* alleles are those identified by Pedro Cano et al. in "Common and well-documented HLA alleles: report of the Ad-Hoc committee of the American society for histocompatiblity and immunogenetics", Human Immunology, Volume 68, Issue 5, May 2007, Pages 392-417. This table contains counts based on all the allele names occurring within a type allele A *rare* allele is an allele that is not characterized as CWD or as G-Code.
List of Rare Alleles by Locus	This table lists the counts of each rare allele by Locus present in the input file
Summary of Ambiguity Counts by Locus	This table lists the number of typed alleles per locus that are ambiguous and not ambiguous
Summary of Typed Alleles by Locus	This table provides the number of typed alleles per locus.
Run statistics	This table provides the actual run-time (in seconds) of the validation process.

3.4.3 Vector Files

For each HLA locus typed input dataset a single SFVT vector file is generated. The vector files are identified with the following name syntax: "**<Task_ID>.<LOCUS_NAME>**.SFVT_vectors.txt" where <LOCUS_NAME> is the IMGT locus name designator, for example, HLA-A and <Task_ID> is the ID assigned to the submitted MHC SFVT analysis task.

The generated SFVT vector files are text files containing tab-separated data arranged in columns. The file format for a vector file generated from an HLA Typing Result Template file is illustrated in Table 9, "Generated Vector File Format For HLA-A" below.

Table 9, Generated Vector File Format for HLA-A

HLA Variant Type Results						…
Column Name	Experiment Sample User-Defined ID*	Hsa_HLA-A_SF1 Allele 1	Hsa_HLA-A_SF1 Allele 2	Hsa_HLA-A_SF2 Allele 1	Hsa_HLA-A_SF2 Allele 2	…
	AB1234	A*0201	A*0101	HAS_HLA-A_SF2_VT6	HAS_HLA-A_SF2_VT1	…
	AB2345	A*0101	No Data	HAS_HLA-A_SF2_VT1	No Data	…
	AB3456	A*0101	Ambiguous Type	HAS_HLA-A_SF2_VT1	Ambiguous Type	…
	AB4567	A*0201	A*0202	HAS_HLA-A_SF2_VT6	Type Unknown	…
	AB5678	Ambiguous Type	Ambiguous Type	Ambiguous Type	Ambiguous Type	…

3.4.4 Generation Summary

The generation summary file**, SFVT.generation_summary_Log.log**, contains the following informational tables occurring in the summary file in the order specified in Table 10, "Informational Tables in Generation Summary". These tables describe the results of the vector file generation process.

Table 10, Informational Tables in Generation Summary

Table Name	Description
Version information	This tables contains the following information: i. Versions of the source data utilized in generating the input alleles ii. Version of HLA QC pipeline iii. Start time of the task
Files and species information	This table provides the species information, the input file type and the list of generated files
Subject Statistics	This table provides the number of subjects that occurred in the file
Summary of Error Counts by Locus	This table provides the number of typed alleles, by locus, that were validated successfully (**No Er**ror) and unsuccessfully (**Erro**r).
Number of Sequence Feature Variant Types (SFVTs) defined per Locus	This table provides the number of sequence feature variant types defined per locus for each data type. The data types are as defined in Table 6. They are: **Ambiguous Type** **Defined Type** **No Data** **Null** **Unknown Type**

Table Name	Description
Summary of Data Types by Locus	This table provides the number of typed alleles of different data types per locus. The data types are: **NULL Allele**—a NULL allele as defined by IMGT **Serological**—a 2-digit allele only specifying the allele's serological category **Missing SFVT Data**—alleles whose variant type information is not yet computed
Summary of Typed Alleles by Locus	This table provides the number of typed alleles per locus.
List of Errors	This table only appears if there were errors in validating the allele data. This table lists all errors that occurred with the following column information: **Row Num**—data row number in the file **Row Id**—the 'subject ID' or the 'Experiment Sample User-Defined ID' **Col Name**—the column name in the input file identifying the locus and the allele **Allele entry Data**—the allele entry **Err Num**—the error number of the error as defined in Appendix B **Error Message**—a short error message
Error and Note Details by entries	This table of information only occurs if there were errors or notes that occurred in the validation process. The error messages that can occur in this table are specified in detail in Appendix B
Category Errors	This table only occurs if there were any errors. It provides the counts of each error by error category. The error categories and error numbers in this table correspond to those listed in Appendix B.
Validation Notes for entries	This table only appears if there were processing notes that occurred in validating the allele data. This table lists all notes that occurred with the following column information: **Row Num**—data row number in the file **Row Id**—the 'subject ID' or the 'Experiment Sample User-Defined ID' **Col Name**—the column name in the input file identifying the locus and the allele **Allele entry Data**—the allele entry **Note Message**—a note about the issue that occurred
Run statistics	This table provides the actual run-time (in seconds) of the validation process.

4.0 HLA QC USING PYPOP

This Section describes the HLA quality control (HLA QC) pipeline using the PyPop tool [Lancaster et al: 2007]. HLA Typing data can be provided in the file formats as specified in Appendix A or generated from HLA typing results data previously uploaded into ImmPort.

Before an HLA typing data can be used as input to PyPop, it is necessary to validate that the typed HLA alleles are in compliance with the latest report of the *Nomenclature for Factors of the HLA System* (http://hla.alleles.org/nomenclature/nomenc_reports.html) (see Sections 3.1 & 3.2). After validation, the validated data is submitted to an allele disambiguation process based on common and well-documented alleles as defined by Pedro Cano et.al in "*Common and well-documented HLA alleles: report of the Ad-Hoc committee of the American society for histocompatiblity and immunogenetics*", Human Immunology, Volume 68, Issue 5, May 2007, Pages 392-417. The disambiguated output is converted into PyPop input format and submitted to PyPop. The documentation on PyPop was obtained from the PyPop tool's user guide that can be downloaded from the following site: www.pypop.org.

A typical PyPop run might take anywhere from a few of minutes to a few hours, depending on how large your data set is and who else is using the system at the same time.

The following files are generated during the HLA QC pipeline:

i. *Allele validation output* – This file is similar to the input HLA typing results file except that only the validated alleles are included and this will be the file used by the disambiguation process (see Section .3.4.1).
ii. *Allele validation summary* – This is a report of the validation on the alleles in the input dataset. It contains information on the number of typed alleles that have errors and other useful information such as the number of common and rare alleles present for each HLA locus typed in the dataset (see Section 3.4.2).
iii. *Allele disambiguation output* – This file is similar to the input HLA typing results file except that the data has been disambiguated (type alleles will be CWD alleles, g-codes, and rare alleles) and will be the input file used by PyPop.
iv. *Allele disambiguation summary* – This is a report of the disambiguation on the alleles in the validated dataset. It contains information on the number of typed alleles that have errors and other useful information such as the number of common and rare alleles present for each HLA locus typed in the dataset
v. *PyPop input file* – The disambiguated data reformatted into PyPop input format.
vi. *PyPop outputs* – Output from a successful PyPop run includes: the complete results in XML format and a human-readable text format of the results.
vii. *PyPop summary* – A report summarizing the results of the PyPop run including the Pypop configuration file.

The following Subsections specify the allele disambiguation process, the PyPop configuration file, and the files output by the run. Errors generated by the pipeline are listed in Appendix B.

4.1 ALLELE DISAMBIGUATION

The allele ambiguity resolution process assumes that the input file has been validated (Sections 3.1 & 3.2). This process assumes that each allele cell is composed of a single allele name, g-code, serological code, or a collection of allele names.

An allele is **common and well-documented (CWD)**, if its name is a common and well-documented allele as defined in the paper by Pedro Cano et.al, "Common and well-documented HLA alleles: report of the Ad-Hoc committee of the American society for histocompatiblity and immunogenetics", Human Immunology, Volume 68, Issue 5, May 2007, Pages 392-417. Also, a four (4) digit allele is **common and well-documented** if its four digits appear as the first four digits of one of the CWD allele names as defined in the paper. A **rare** allele is an allele that is not a CWD as defined above. Furthermore, an allele is a member of a gcode if it is defined as such in the above paper. A gcode group is a mechanism to group alleles that have the same sequence at the peptide level for exons 2 & 3, for Class I loci, or exon 2, for Class II loci. Also, a four (4) digit allele is a member of a gcode if it appears as the first four digits of an allele that is defined to reside in the gcode as defined in the paper.

In the allele ambiguity resolution process, cells containing only rare allele(s) will be left 'as-is'. That is, all the alleles are reduced to their four (4) digit equivalents and written out to the ambiguity resolution file. Also, a note will be generated to the logging file if there is more than one rare allele indicating a cell contains only rare alleles. The log file does not register a note if the cell consists of a single rare allele.

The following decision process specifies the allele ambiguity resolution using the names as presented in the input file

The following conditions are assumed.

1. The data in the input file has been validated using the current IMGT allele dataset.
2. Consider only non-trivial cells from the file for each locus. A cell is non-trivial if it contains at least one allele entry.
3. The alleles for the given entry (and locus) are presented as the **names** list, (N_1, N_2, N_3, ...) (See table 11), derived directly from the file.
4. If there is only one allele name entry, then the name N_1 can be an allele name, a gcode, or serological value (2-digit code). Otherwise, for multiple **names** in a single HLA type entry, the list will consist of only allele names.

For each name N_i in the **names**, the following set of attributes as defined in the Table 11, "Attributes Defined for Each Name N_i", below are computed for it.

Table 11, Attributes Defined for Each Name N_i

Attribute	Definition	Comments
N_i.type	Type of the name: **allele**, **gcode**, or **sero**	• **gcode** is a name gcode group name (one ending in 'g') • **sero** is a serological (2-digit) code • **allele** is neither of the above
N_i.dallele	• If type is **gcode**, then the full name without the locus name but including the 'g' suffix, • If type is **sero**, then the 2-digit abbreviation • If type is **allele**, the 4-digits (peptide) abbreviation	
N_i.fallele	Name without the locus name, but including the (optional) suffix from the input file	Suffixes like the gcode suffix 'g', or allele suffixes 'N', 'L', etc.
N_i.cwd	• If type is **allele**, a Boolean indicating whether allele is a CWD allele (TRUE) or not (FALSE) • If type is **gcode** or **sero**, the value is FALSE.	The CWD designation for an **allele** is determined as follows: • If the name N_i is only 4-digits without a suffix, then the CWD designation is determined N_i.dallele (same as N_i in this case) • If the name has more than 4-digits and/or a suffix, then the CWD designation is determined using N_i.fallele.
N_i.gcode	• If type is **allele**, the gcode group name without locus name into which the allele name is grouped (see comments for details); if there is no group code, then the attribute is empty. • If type is **gcode**, then the attribute has the value N_i.fallele.	• For type **allele** and the name N_i is only 4-digits without suffix, the gcode is determined using only 4-digit lookup. • For type **allele** and the name has more than 4-digits and/or a suffix, then the gcode is determined using a full allele name lookup using N_i.

The following processing cases are considered as defined in the following Table 12, "Processing Cases".

Table 12, Processing Cases

Processing Case	Definition
(==1)	Only one name is entered for an HLA type entry
(>1)	Multiple names are entered for an HLA type entry

(==1) Processing Case:

The decision tree is defined in the Table 13, "(==1) Decision Tree", below. The Condition and Sub-Condition are considered in priority order. That is only one condition and optionally one subsequent Sub-Condition is executed for each N_1.

Table 13, (==1) Decision Tree

Condition	Sub-Condition	Result
N_1.type in {'sero', 'gcode'}		return N_1.dallele
N_1.type == 'allele'	N_1.gcode defined	return N_1.gcode
	N_1.cwd is FALSE	return N_1.dallele
	N_1.cwd is TRUE and N_1.fallele is a null allele (N-suffix)	return N_1.fallele
	N_1.cwd is TRUE and N_1.fallele has more than 4-digits	Determine gcode using N_1.dallele; if gcode exists return it, otherwise return N_1.dallele

(>1) Processing Case:

This processing case is defined by two steps, the binning process, and result determination process. Recall that in this case all names N_i is type **allele**.

1. Binning Process

In this step, the names N_i are binned into the following lists defined in the Table 14, "Binning Lists", below:

Table 14, Binning Lists

List Name	Definition
cwds	List of unique names for which N_i.cwd is TRUE, but no gcode can be determined for it (see the decision table below)
rares	List of unique names for which N_i.cwd is FALSE
gcodes	list of unique gcodes determined for names for which N_i.cwd is TRUE (see decision table below)

For each name N_i in the names list, the decision tree specified in the following Table 15, "(>1) Decision Tree", defines how N_i is binned. The Condition and Sub-Condition are considered in priority order.

Table 15, (>1) Decision Tree

Condition	Sub-Condition	Result
N_i.cwd is FALSE		bin N_i.dallele into rares
N_i cwd is TRUE	N_i.gcode defined	bin N_i.gcode into gcodes
	N_i.cwd is TRUE and N_i.fallele is a null allele (N-suffix)	bin N_i.fallele into cwds
	N_i.cwd is TRUE and N_i.fallele has more than 4-digits	Determine gcode using N_i.dallele; if gcode exists return it, otherwise return N_i.dallele

2. Result Determination Process

The decision tree for determining the resulting cell is defined in the following Table 16, "Cell Results".

Table 16, Cell Results

Condition	Cell Result
rares > 0 and cwds == 0 and gcodes == 0	return rares
cwds > 0 and gcodes == 0	return cwds
cwds == 0 and gcodes > 0	return gcodes

Condition	Cell Result
cwds > 0 and gcodes > 0	return gcodes union cwds

Table 17, "Validated and Ambiguity Resolution for HLA-A Alleles", illustrates the validation and the ambiguity resolution processing results for locus HLA-A.

Table 17, Validated, and Ambiguity Resolution for HLA-A Alleles

Original Cell	Validated Cell	Ambiguity Resolved Cell
010101g/A*0101		
0110	**0110**	**0110**
0110/0106	**0110/0106**	**0106/0110**
0110/106		
02	**02**	**02**
0209/43N	**0209/0243N**	**020101g**
0294N	**0294N**	**0294**
02AMJM/A*0101		
03	**03**	**03**
03/02		
0300	**03**	**03**
0300/02		
1010.0		
1010102N	**01010102N**	**010101g**
110	**0110**	**0110**
110/0106	**0110/0106**	**0106/0110**
110/06	**0110/0106**	**0106/0110**
110/106		
2	**02**	**02**
2202		
2402101	**24020101**	**240201g**
294N	**0294N**	**0294**
3	**03**	**03**
3/02		
300	**03**	**03**
300/02		
3013	**3013**	**3013**
5101/17/21		
68011/0101	**680101/0101**	**010101g/680101g**
68011/2402101	**680101/24020101**	**240201g/680101g**
A*0101.1	**0101**	**010101g**
A*0101.1N		

Original Cell	Validated Cell	Ambiguity Resolved Cell
A*0101/0200		
A*0101/02AMJM		
A*0101/A*0101011g		
A*0101/A*010101g		
A*0101/A*02		
A*0101/A*0200		
A*0101/A*0212AMJM		
A*0101/A*102		
A*0101/A*3		
A*0101/A*B03		
A*0101011g		
A*0101011g/A*0101		
A*010101g	010101g	010101g
A*010101g/A*0101		
A*010102g		
A*0110	0110	0110
A*0110/0106	0110/0106	0106/0110
A*02	02	02
A*0200N		
A*020101g	020101g	020101g
A*02011	020101	020101g
A*020120	020118	020101g
A*0212AMJM		
A*0212AMJM/A*0101		
A*021AMJM		
A*02202	022002	0220
A*0294N	0294N	0294
A*02AMJM	0201/0209/0243N/0266	020101g
A*02AMJM/A*0101		
A*02BRHJ	0201/0209/0243N/0266/0275/0283N/0289	020101g
A*02N		
A*03	03	03
A*03/02		
A*0300	03	03
A*0300/02		
A*03013	030103	030101g
A*0312345678		
A*03BRHJ		
A*03VS	0301/0320	030101g
A*101/A*102		
A*10101g		
A*10102g		

Original Cell	Validated Cell	Ambiguity Resolved Cell
A*110		
A*2		
A*200N		
A*20102	020102	020101g
A*2202		
A*2312/14/12/14	2312/2314	2312/2314
A*2312/14/15	2312/2314/2315	2312/2314/2315
A*2401		
A*2402101/02L	24020101/24020102L	240201g
A*24022	240202	240201g
A*2901102N	29010102N	2901g
A*29011N		
A*294N		
A*2N		
A*3		
A*3		
A*3/02		
A*3/A*0101		
A*300		
A*300/02		
A*3013	3013	3013
A*3021	301102	3011
A*312345678		
A*B03		
A*B03/A*0101		
A*11XX	11	11
A*11xx	11	11
11XX	11	11
11xx	11	11
A*11XX/A*2301		
11XX/2301		
A*2301/A*11XX		
2301/11XX		
A*11xx/A*2301		
11xx/2301		
A*2301/A*11xx		
2301/11xx		
A*0102	0102	0102
0102	0102	0102
102	0102	0102
0102/0106/0103/0110	0102/0106/0103/0110	0102/0103
0201, 0209, 0243N, 0266	0201/0209/0243N/0266	020101g

Original Cell	Validated Cell	Ambiguity Resolved Cell
A*0102/A*0103/A*2612	0102/0103/2612	0102/0103/2612
A*0104N/A*02010101	0104N/02010101	010101g/020101g
01010102N/010104/0117/020107	01010102N/010104/0117/020107	0101/0117/0201
010105	010105	010101g
A*020170		
A*260101/2624/2626	2601g	2601g
A*02G1	020101g	020101g
A*7401/7402	7401g	7401g
A*1101/1121N	110101g	110101g

4.2 PyPop Configuration File

The population genetic analyzes that are run on your disambiguated data and the manner in which the data is interpreted by PyPop is controlled by a configuration file. The content of the ini-configuration appear in the Pypop summary file, "***Task#.Allele.pypop_summary_Log.log***". The ini-configuration file is a text property file consisting of comments (which are lines that start with a semi-colon), sections (lines with labels in square brackets), and options (lines specifying settings relevant to that section in the option=value format).

```
[General]                          ❶
debug=0

[ParseGenotypeFile]                ❷
untypedAllele=****
alleleDesignator=*
validSampleFields=*a_1
 *a_2
 *c_1
 *c_2
 *b_1
 *b_2

[HardyWeinberg]                    ❸
lumpBelow=5

[HardyWeinbergGuoThompson]         ❹
dememorizationSteps=2000
samplingNum=1000
samplingSize=1000

[HomozygosityEWSlatkinExact]       ❺
numReplicates=10000

[Emhaplofreq]                      ❻
allPairwiseLD=1
allPairwiseLDWithPermu=0
;;numPermuInitCond=5
```

Figure 8 Sample Ini-Configuration File

The configuration file sections are described below:

1. [General]
   This Section contains variables that control the overall behavior of PyPop.

   - *debug=0*: This setting is for debugging. Setting it to 1 will set off a large amount of output of no interest to the general user.
   - *outFilePrefixType*: The default is set as filename, which will result in three output files [Default: filename]

2. Specifying data formats:
   There are two possible section formats: [ParseGenotypeFile] and [ParseAlleleCountFile. If your data is genotype data, the Section will be labeled: [ParseGenotypeFile].

   - *alleleDesignator*: This option is used to tell PyPop what is allele data and what isn't. The default is *.
   - *popNameDesignator*: There is a special designator to mark the population name field, which is usually the first field in the data block. [Default: +]
     If you are analyzing data that contains a population name for each sample, then the first entry in your *validSampleFieldsSection* needs the prefix '+'
   - *validPopFields:* [Default:] Required when a population data block is present in PyPop input data file.
   - *validSampleFields*: This optional field contains the names of the loci immediately preceding your genotype data. The first line (*validSampleFields=*) consists of the name of your sample field (if it contains allele data, the name of the field is preceded by the character designated in the *alleleDesignatoroption* above).
     All subsequent lines after the first must be preceded by one space (again if it contains allele data, the name of the field is preceded by the character designated in the *alleleDesignatoroption* above).

```
validSampleFields=*a_1
 *a_2
 *c_1
 *c_2
 *b_1
 *b_2    Note initial space at start of line.
```

```
validSampleFields=populat
 id
 *a_1
 *a_2
 *c_1
 *c_2
 *b_1
 *b_2
```

Figure 9 Here is example that includes identifying (no allele data) information such as sample id (id) and population name (populat)

3. [HardyWeinberg]
   Hardy-Weinberg analysis is enabled by the presence of this Section.

   - *lumpBelow*: This option value represents a cutoff value. Alleles with an expected value equal to or less than lumpBelow will be lumped together into a single category for the purpose of calculating the degrees of freedom and overall p-value for the chi-squared Hardy Weinberg test.

4. [Emhaplofreq]
   The presence of this Section enables haplotype estimation and calculation of linkage disequilibrium (LD) measures.

   - *allPairwiseLD*: Set this to 1(one) if you want the program to calculate all pair-wise LD for your data, otherwise set this to 0(zero).
   - *allPairwiseLDWithPermu*: Set this to a positive integer greater than 1 if you need to determine the significance of the pair wise LD measures in the previous Section. The number you use is the number of permutations that will be run to ascertain the significance (this needs to be at least 1000 or greater). (Note this is done via permutation testing performed after the pair wise LD test for all pairs of loci. Note also that this test can take DAYS if your data is highly polymorphic.)
   - *permutationPrintFlag*: Determines whether the likelihood ratio for each permutation will be logged to the XML output file, this is disabled by default. [Default: 0(OFF)].
   - *lociToEstHaplo*: In this option you can list the multi-locus haplotypes for which you wish the program to estimate and to calculate the LD. It is a comma-separated list of colon joined loci.
   - *numPermuInitCon:* Set this to change the number of initial conditions used per permutation. [Default: 5]. (Note: this parameter is only used if *allPairwiseLDWithPermu* is set and nonzero).

4.3 HLA QC OUTPUTS

The HLA QC pipeline creates the following files which are described in the Subsections below:

- **validated file (see Section 3.4.1)**
- **validation summary (see Section 3.4.2)**
- **disambiguated file**
- **disambiguation summary**
- **PyPop input data file**
- **PyPop result files**
- **Pypop summary**

4.3.1 Disambiguated File

The validated data (see Section 3.4.1) is disambiguated as described in Section 4.1. A file is generated as a result of the disambiguation. The file contains: CWD alleles, g-codes, rare alleles, and serological alleles. This file has the same format as the file input to HLA QC pipeline and is identified by the following name syntax: "**<Task_ID>**.allele.disambiguation_output.txt" where <Task_ID> is the ID assigned to the task running the HLA QC pipeline. The task IDs along with other related information are listed in the analysis history page. However, the allele content has been modified to represent the validated alleles as shown for few examples in Table 17. An example of a validated file is illustrated in Table 18, "Example of a Disambiguated File" below.

Table 18, Example of a Disambiguated File

HLA Typing Results												…
Please do not delete or edit this column												…
Column Name	Experiment Sample User-Defined ID*	HLA-A Allele 1	HLA-A Allele 2	HLA-B Allele 1	HLA-B Allele 2	HLA-C Allele 1	HLA-C Allele 2	HLA-DPA1 Allele 1	HLA-DPA1 Allele 2	HLA-DPB1 Allele 1	HLA-DPB1 Allele 2	…
	AB1234	020101g	010101g	440201g	080101g	050101g	070101g			0401	0401	…
	AB2345	010101g		080101g	440201g	070101g	050101g			0201	0201	…
	AB3456	010101g	240201g/240301g	080101g	3502	070101g	040101g			0402g	0401	…
	AB4567	020101g	0202	4901	080101g	070101g	070101g			030101g	0401	…
	AB5678	030101g	020101g	2702	440201g	0202	050101g			0402g	0402g	….

Note that, the empty cells are replaced with an empty column, and allele names are presented numerically (no prefix) in the standard CWD/IMGT format.

4.3.2 Disambiguation Summary

The validation summary file, **Allele.disambiguation_summary_Log.log**, contains the following informational tables occurring in the summary file in the order specified in Table 19, "Informational Tables in Disambiguation Summary". These tables describe the results of the disambiguation process.

Table 19, Informational Tables in Disambiguation Summary

Table Name	Description
Version information	This tables contains the following information: i. Versions of the source data utilized in validating the input alleles ii. Version of the vector file generator tool iii. Start time of the task
Files and species information	This table provides the species information, the input file type and the list of generated files
Subject Statistics	This table provides the number of subjects present in the input data
Summary of Error Counts by Locus	This table provides the number of typed alleles, by locus, that were validated successfully (**No Er**ror) and unsuccessfully (**Erro**r).
List of Errors	This table only appears if there were errors in validating the allele data. This table lists all errors that occurred with the following column information: **Row Num**—data row number in the file **Row Id**—the 'subject ID' or the 'Experiment Sample User-Defined ID' **Col Name**—the column name in the input file identifying the locus and the allele **Allele entry Data**—the allele entry **Err Num**—the error number of the error as defined in Appendix B **Error Message**—a short error message
Validation Notes for entries	This table only appears if there were processing notes that occurred in validating the allele data. This table lists all notes that occurred with the following column information: **Row Num**—data row number in the file **Row Id**—the 'subject ID' or the 'Experiment Sample User-Defined ID' **Col Name**—the column name in the input file identifying the locus and the allele **Allele entry Data**—the allele entry **Note Message**—a note about the issue that occurred
Error and Note Details by entries	This table of information only occurs if there were errors or notes that occurred in the validation process. The error messages that can occur in this table are specified in detail in Appendix B

Table Name	Description
Category Errors	This table only occurs if there were any errors. It provides the counts of each error by error category. The error categories and error numbers in this table correspond to those listed in Appendix B.
Summary of Typed Alleles by Locus	This table provides the number of typed alleles per locus.
Run statistics	This table provides the actual run-time (in seconds) of the validation process.

4.3.3 Pypop Input Data File

The ImmPort's HLA QC pipeline generates the input data file for the PyPop tool from the disambiguated output specified in Section 4.3.1. The input to the PyPop tool is either as genotypes, or in an allele count format, depending on the format of your data. The filename for the input file generated for the PyPop tool is in the following format: "***Task#.allele.pypop.txt***".

Note that for genotype data, each locus corresponds to two columns in the Pypop input file. The locus name is repeated with a suffix such as _1, _2 (the default). Although PyPop needs this distinction to be made, phase is NOT assumed, and if known it is ignored.

```
populat    id         a_1   a_2   c_1   c_2   b_1   b_2
UchiTelle  UT900-23   ****  ****  0102  02025 1301  18012
UchiTelle  UT900-24   0101  0201  0307  0605  1401  39021
UchiTelle  UT900-25   0210  03012 0712  0102  1520  1301
UchiTelle  UT900-26   0101  0218  0804  1202  35091 4005
UchiTelle  UT910-01   2501  0201  1507  0307  51013 1401
UchiTelle  UT910-02   0210  3204  1801  0102  78021 1301
UchiTelle  UT910-03   03012 3204  1507  0605  51013 39021
```

Figure 10 Multi-locus allele-level HLA genotype data with sample information

This example shows a data file that has non-allele data in some columns; namely, population (populat) and sample identifiers (id).

The default value for untyped or missing data is a series of four asterisks (****) as specified by the Pypop ini-configuration file.

For individuals who were not typed at all loci, the data in loci for which they are typed will be used on all single-locus analyzes for that individual and locus, so that you see the value of the number of individuals (n) vary from locus to locus in the output. These individuals' data will also be used for multi-locus analyzes. Only the loci that contain no missing data will be included in any multi-locus analysis.

If an individual is only partially typed at a locus, it will be treated as if it was completely untyped, and data for that individual for that locus will be dropped from all analyzes.

4.3.4 Pypop Result Files

A successful PyPop run will produce two output files: "***Task#.allele.pypop-out.txt***" and "***Task#.allele.pypop-out.xml***". The XML file, "***Task#.allele.pypop-out.xml***", is the primary result output created by PyPop, and the human-readable file, "***Task#.allele.pypop-out.txt***", is a text summary of the results. This text summary is discussed below.

4.3.4.1 Population Summary

A population summary is generated for each dataset analyzed. This summary provides basic demographic information and summarizes information about the sample size.

```
Population Summary
==================
Population Name: UchiTelle
       Lab code: USAFEL
  Typing method: 12th Workshop SSOP
      Ethnicity: Telle
      Continent: NW Asia
Collection site: Targen Village
       Latitude: 41 deg 12 min N
      Longitude: 94 deg 7 min E

Population Totals
_________________
Sample Size (n): 47
Allele Count (2n): 94
Total Loci in file: 9
Total Loci with data: 8
```

Figure 11 Sample output for the population summary

4.3.4.2 Single Locus Analyzes

Basic Allele Count Information

Information relevant to individual loci is reported. Sample size and allele counts will differ among loci if not all individuals were typed at each locus. Untyped individuals are those for which one or two alleles were not reported. The alleles are listed in descending frequency (and count) in the left hand column, and are sorted numerically in the right column. The number of distinct alleles 'k' is reported.

```
I. Single Locus Analyses
========================

1. Locus: A
___________

1.1. Allele Counts [A]
----------------------
Untyped individuals: 2
Sample Size (n): 45
Allele Count (2n): 90
Distinct alleles (k): 10

Counts ordered by frequency   | Counts ordered by name
Name      Frequency (Count)   | Name      Frequency (Count)
0201      0.21111   19        | 0101      0.13333   12
0301      0.15556   14        | 0201      0.21111   19
0101      0.13333   12        | 0210      0.10000   9
2501      0.12222   11        | 0218      0.10000   9
0210      0.10000   9         | 0301      0.15556   14
0218      0.10000   9         | 2501      0.12222   11
3204      0.08889   8         | 3204      0.08889   8
6901      0.04444   4         | 6814      0.03333   3
6814      0.03333   3         | 6901      0.04444   4
7403      0.01111   1         | 7403      0.01111   1
Total     1.00000   90        | Total     1.00000   90
```

Figure 12 Single locus analyzes – basic locus information sample output

In the cases where there is no information for a locus, a message is displayed indicating lack of data.

Chi-square Test for Deviation from Hardy-Weinberg Proportions (HWP)

For each locus, the observed genotype counts are compared to those expected under Hardy Weinberg proportions (HWP). A triangular matrix reports observed and expected genotype counts. If the matrix is more than 80 characters, the output is split into different Sections. Each cell contains the observed and expected number for a given genotype in the format observed/expected.

```
6.2. HardyWeinberg [DQA1]
-------------------------
Table of genotypes, format of each cell is: observed/expected.

0201 8/5.1
0301 4/4.0 1/0.8
0401 3/6.9 1/2.7 6/2.3
0501 8/9.9 5/3.8 5/6.7 6/4.8
      0201   0301   0401   0501
                            [Cols: 1 to 4]
```

Figure 13 Single locus analyzes – Sample output of Hardy-Weinberg genotype table

The values in this matrix are used to test hypotheses of deviation from HWP. The output also includes the chi-square statistic, the number of degrees of freedom and associated p-value for a number of classes of genotypes and is summarized in the following table:

	Observed	Expected	Chi-square	DoF	p-value	
Common	N/A	N/A	4.65	1	0.0310*	❶
Lumped genotypes	N/A	N/A	1.17	1	0.2797	❷
Common + lumped	N/A	N/A	5.82	1	0.0158*	❸
All homozygotes	21	13.01	4.91	1	0.0268*	❹
All heterozygotes	26	33.99	1.88	1	0.1706	❺
Common heterozygotes by allele						❻
0201	15	20.78	1.61		0.2050	
0301	10	10.47	0.02		0.8850	
0401	9	16.31	3.28		0.0703	
0501	18	20.43	0.29		0.5915	
Common genotypes						❼
0201:0201	8	5.11	1.63		0.2014	
0201:0401	3	6.93	2.23		0.1358	
0201:0501	8	9.89	0.36		0.5472	
0401:0501	5	6.70	0.43		0.5109	
Total	24	28.63				

Figure 14 Single locus analyzes – Sample output of Hardy-Weinberg genotype classes

Explanation of each genotype class

1. *Common:*
   The result for goodness of fit to HWP using only the genotypes with at least *lumpBelow* expected counts (the common genotypes) (in the output shown throughout this example *lumpBelow* is equal to 5).

   If the dataset contains no genotypes with expected counts equal or greater than *lumpBelow*, then there are no common genotypes and the following message is reported: “No common genotypes; chi-square cannot be calculated”

   The analysis of common genotypes may lead to a situation where there are fewer classes (genotypes) than allele frequencies to estimate. This means that the analysis cannot be performed (degrees of freedom < 1). In such a case, the following message is reported that explains why the analysis cannot be performed: “Too many parameters for chi-square test”

   To obviate this as much as possible, only alleles which occur in common genotypes are used in the calculation of degrees of freedom.

2. *Lumped genotypes:*
   The result for goodness of fit to HWP for the pooled set of genotypes that individually have less than *lumpBelow* expected counts.

The pooling procedure is designed to avoid carrying out the chi-square goodness of fit test in cases where there are low expected counts, which can lead to spurious rejection of HWP. However, in certain cases it may not be possible to carry out this pooling approach. The interpretation of results based on lumped genotypes will depend on the particular genotypes that are combined in this class.

If the sum of expected counts in the lumped class does not add up to *lumpBelow*, then the test for the *lumped genotypes* cannot be calculated and the following message is reported: "The total number of expected genotypes is less than 5"

This may by remedied by combining rare alleles and recalculating overall chi-square value and degrees of freedom. (This requires appropriate manipulation of the data set by hand and is not a feature of PyPop).

3. *Common + lumped:*
   The result for goodness of fit to HWP for both the common and the lumped genotypes.
4. *All homozygotes:*
   The result for goodness of fit to HWP for the pooled set of homozygous genotypes.
5. *All heterozygotes:*
   The result for goodness of fit to HWP for the pooled set of heterozygous genotypes.
6. *Common heterozygotes:*
   The *common heterozygote by allele Section summarizes* the observed and expected number of counts of all heterozygotes carrying a specific allele withexpected value $\geq$ *lumpBelow*.
7. *Common genotypes:*
   The *common genotypes* by genotype Section lists observed, expected, chi-square and p-values for all observed genotypes with expected values $\geq$ *lumpBelow.*

4.3.4.3 Mulit-locus Analyzes

Haplotype frequencies are estimated using the iterative Expectation-Maximization (EM) algorithm ([Dempster:1977]; [Excoffier:Slatkin:1995]). Multiple starting conditions are used to minimize the possibility of local maxima being reached by the EM iterations. The haplotype frequencies reported are those that correspond to the highest logarithm of the sample likelihood found over the different starting conditions and are labeled as the maximum likelihood estimates (MLE).

The output provides the names of loci for which haplotype frequencies were estimated, the number of individual genotypes in the dataset (before-filtering), the number of genotypes that have data for all loci for which haplotype estimation will be performed (after-filtering), the number of unique phenotypes (unphased genotypes), the number of unique phased genotypes, the total number of possible haplotypes that are compatible with the genotypic data (many of

these will have an estimated frequency of zero), and the log-likelihood of the observed genotypes under the assumption of linkage equilibrium.

All Pairwise LD

A series of linkage disequilibrium (LD) measures are provided for each pair of loci. We report two measures of overall linkage disequilibrium. D' [Hedrick:1987] weights the contribution to LD of specific allele pairs by the product of their allele frequencies; Wn [Cramer:1946] is a re-expression of the chi-square statistic for deviations between observed and expected haplotype frequencies. Both measures are normalized to lie between zero and one.

D' Overall LD, summing contributions (D'ij = Dij / Dmax) of all the haplotypes in a multi-allelic two locus system, can be measured using Hedrick's D' statistic, using the products of allele frequencies at the loci, p_i and q_j, as weights.

$$D' = \sum_{i=1}^{I} \sum_{j=1}^{J} p_i q_j \left| D'_{ij} \right|$$

II. Multi-locus Analyses
========================

Haplotype/ linkage disequlibrium (LD) statistics

Pairwise LD estimates

Locus pair	D'	Wn	ln(L_1)	ln(L_0)	S	# permu	p-value
A:C	0.49229	0.39472	-289.09	-326.81	75.44	1000	0.8510
A:B	0.50895	0.40145	-293.47	-330.83	74.73	1000	0.8730
A:DRB1	0.44304	0.37671	-282.00	-309.16	54.32	1000	0.7540
A:DQA1	0.29361	0.34239	-257.94	-269.88	23.88	1000	0.9020
A:DQB1	0.39266	0.37495	-275.58	-297.61	44.07	1000	0.8140
A:DPA1	0.31210	0.37987	-203.89	-206.99	6.21	1000	0.8840
A:DPB1	0.42241	0.40404	-237.84	-262.05	48.42	1000	0.5930
C:B	0.88739	0.85752	-210.36	-342.68	264.63	1000	0.0000***
C:DRB1	0.48046	0.47513	-280.34	-317.65	74.62	1000	0.2140
C:DQA1	0.42257	0.49869	-250.36	-276.72	52.73	1000	0.0370*
C:DQB1	0.45793	0.49879	-269.54	-305.27	71.46	1000	0.0580
C:DPA1	0.37214	0.46870	-208.99	-215.36	12.74	1000	0.7450
C:DPB1	0.46436	0.36984	-242.45	-268.45	52.01	1000	0.6290
B:DRB1	0.50255	0.41712	-286.79	-320.50	67.42	1000	0.4140
B:DQA1	0.41441	0.42844	-259.86	-279.56	39.40	1000	0.3880
B:DQB1	0.49040	0.43654	-277.29	-308.12	61.65	1000	0.2870
B:DPA1	0.29272	0.38831	-213.43	-218.01	9.14	1000	0.8780
B:DPB1	0.46082	0.38001	-247.83	-272.77	49.86	1000	0.7320
DRB1:DQA1	0.91847	0.91468	-164.06	-254.54	180.96	1000	0.0000***
DRB1:DQB1	1.00000	1.00000	-147.73	-283.09	270.72	1000	0.0000***

...

Figure 15 Multilocus analyzes – Sample output of all pairwise LD

Wn Also known as Cramer's V Statistic [Cramer:1946], Wn, is a second overall measure of LD between two loci. It is a re-expression of the Chi-square statistic, XLD2, normalized to be between zero and one.

$$W_n = \left[\frac{\sum_{i=1}^{I} \sum_{j=1}^{J} D_{ij}^2 / p_i q_j}{\min(I-1, J-1)} \right]^{\frac{1}{2}} = \left[\frac{X_{LD}^2 / 2N}{\min(I-1, J-1)} \right]^{\frac{1}{2}}$$

When there are only two alleles per locus, Wn is equivalent to the correlation coefficient between the two loci, defined as $r = \sqrt{D_{11}/(p_1 p_2 q_1 q_2)}$.

For each locus pair the log-likelihood of obtaining the observed data given the inferred haplotype frequencies [ln(L_1)], and the likelihood of the data under the null hypothesis of linkage equilibrium [ln(L_0)] are given. The statistic S is defined as twice the difference between these likelihoods. S has an asymptotic chi-square distribution, but the null distribution of S is better approximated using a randomization procedure. The empirical distribution of S is generated by shuffling genotypes among individuals, separately for each locus, thus creating linkage equilibrium (# permu indicates how many permutations were carried out). The p-value is the fraction of permutations that results in values of S greater or equal to that observed. A p-value < 0.05 is indicative of overall significant LD. Individual LD coefficients, Dij, are stored in the XML output file, but are not printed in the default text output.

Haplotype frequency estimation

```
Haplotype frequency est. for loci: A:B:DRB1
-------------------------------------------
Number of individuals: 47 (before-filtering)
Number of individuals: 45 (after-filtering)
Unique phenotypes: 45
Unique genotypes: 113
Number of haplotypes: 188
Loglikelihood under linkage equilibrium [ln(L_0)]: -472.700542
Loglikelihood obtained via the EM algorithm [ln(L_1)]: -340.676530
Number of iterations before convergence: 67
```

Figure 16 Multi-locus analyzes – Sample output of haplotype estimation parameters

The estimated haplotype frequencies are sorted alphanumerically by haplotype name (left side), or in decreasing frequency (right side). Only haplotypes estimated at a frequency of 0.00001 or larger are reported. The first column gives the allele names in each of the three loci, the second column provides the maximum likelihood estimate for their frequencies, (frequency), and the third column gives the corresponding approximate number of haplotypes (#copies).

```
Haplotypes sorted by name                | Haplotypes sorted by frequency
haplotype          frequency # copies    | haplotype          frequency # copies
0101:1301:0402:    0.02222   2.0         | 0201:1401:0402:    0.03335   3.0
0101:1301:1101:    0.01111   1.0         | 3204:1401:0802:    0.03333   3.0
0101:1401:0901:    0.01111   1.0         | 0301:1401:0407:    0.03333   3.0
0101:1520:0802:    0.01111   1.0         | 0301:1301:0402:    0.03333   3.0
0101:1801:0407:    0.01111   1.0         | 0201:1401:1101:    0.03332   3.0
0101:3902:0404:    0.01111   1.0         | 0301:1520:0802:    0.02222   2.0
0101:3902:1602:    0.01111   1.0         | 0101:4005:0802:    0.02222   2.0
0101:4005:0802:    0.02222   2.0         | 0301:3902:0402:    0.02222   2.0
0101:8101:0802:    0.01111   1.0         | 0201:1301:1602:    0.02222   2.0
0101:8101:1602:    0.01111   1.0         | 0218:1401:0404:    0.02222   2.0
0201:1301:1602:    0.02222   2.0         | 0210:5101:1602:    0.02222   2.0
0201:1401:0402:    0.03335   3.0         | 0218:1401:1602:    0.02222   2.0
0201:1401:0404:    0.01111   1.0         | 0101:1301:0402:    0.02222   2.0
0201:1401:0407:    0.02222   2.0         | 2501:4005:0802:    0.02222   2.0
0201:1401:0802:    0.01111   1.0         | 2501:1301:0802:    0.02222   2.0

...
```

Figure 17 Multi-locus analyzes – Sample output of estimated haplotype frequencies

4.3.5 Pypop Summary

The Pypop summary file, **Allele.pypop_summary_Log.log**, contains the following informational tables occurring in the summary file in the order specified in Table 20, "Informational Tables in PyPop Summary". These tables describe the results of the PyPop run.

Table 20, Informational Tables in PyPop Summary

Table Name	Description
Version information	This tables contains the following information: i. Versions of the source data utilized in validating the input alleles ii. Version of the vector file generator tool iii. Start time of the task
PyPop Configuration (ini) File	Contents of PyPop ini-configuration files (see Section 4.2)
Results of PyPop Data Analysis	Pypop analysis results read from the XML result file, Task#.allele.pypop-out.xml, (see Section 4.3.4)
Run statistics	This table provides the actual run-time (in seconds) of the validation process.

5.0 REFERENCES

[Cano:et al:2007] Pedro Cano, "Common and well documented HLA alleles: report of the ad hoc committee of the American Society for Histocompatiblity and Immunogenetics", 2007, 68, 392-417. Human Immunology.

[Cramer:1946] H Cramer, Mathematical Models of Statistics, 1946, Princeton University Press, Princeton NJ.

[Dempster:1977] A Dempster, N Laird, and D Rubin, "Maximum likelihood estimation from incomplete data using the EM algorithm", 1977, 39, 1-38. J Royal Stat Soc.

[Excoffier:Slatkin:1995] Laurent Excoffier and Montgomery Slatkin, "Maximum likelihood estimation of molecular haplotype frequencies in a diploid population", Molecular Biology and Evolution, 1995, 12, 921-927.

[Hedrick:1987] P W Hedrick, "Gametic disequilibrium measures: proceed with caution", 1987, 117, 331-41. Genetics.

[Lancaster:etal:2007b] Alex Lancaster, Richard M Single, Owen D Solberg, Mark P Nelson, and Glenys Thomson, "PyPop update—a software pipeline for large-scale multilocus population genomics", Tissue Antigens, 2007b, 69 Suppl 1, 192-197. [PDF (150 kB)]

APPENDIX A HLA FILE CONTENT FORMATS

This Appendix illustrates acceptable formats for the HLA Typing Result data file formats. Currently, the formats include: ImmPort HLA Typing Result template.

A.1 HLA TYPING RESULTS TEMPLATE

The HLA Typing Result Template format is prescribed by the ImmPort upload system as the format for accepting HLA typing results. The format is illustrated in Table A.1, "HLA Typing Result Template Format". The template uses **" Allele 1**" and **" Allele 2**" to refer to the allele types at the homologous chromosomes.

Table A.1, HLA Typing Result Template Format

HLA Typing Results																	
Please do not delete or edit this column																	
Column Name	Experiment Sample User-Defined ID*	HLA-A Allele 1	HLA-A Allele 2	HLA-B Allele 1	HLA-B Allele 2	HLA-C Allele 1	HLA-C Allele 2	HLA-DRB1 Allele 1	HLA-DRB1 Allele 2	HLA-DPA1 Allele 1	HLA-DPA1 Allele 2	HLA-DPB1 Allele 1	HLA-DPB1 Allele 2	HLA-DQA1 Allele 1	HLA-DQA1 Allele 2	HLA-DQB1 Allele 1	HLA-DQB1 Allele 2
	3600	-	-	-	-	-	-	0101	0301	-	-	-	-	0101	0501	0501	0201
	3601	-	-	-	-	-	-	0701	1501	-	-	-	-	0102	0201	0602	0201

A.2 CUSTOM HLA TYPING RESULTS

The HLA Typing Results can be uploaded using the user's custom formats as specified in Table A.2, "Custom-HLA Typing Result Template Format". The following format rules need to be observed:

i. The first row contains the following name in the first column: "**Custom HLA typing file with phenotypes**"

ii. First column of the header row contains the record IDs and is named "**ID**"

iii. Allele data columns are indicated by the full HLA gene name, e.g. **HLA-DRB1**

iv. The template uses the suffixes "**a**" and "**b**" to refer to the allele types at the homologous chromosomes

v. The rest of the columns are interpreted as the phenotype columns

Table A.2, Custom-HLA Typing Result Template Format

Custom HLA typing file with phenotypes									
ID	Gender	Status	Race	Skin_Type	ANA	HLA-DRB1a	HLA-DRB1b	HLA-DQA1a	HLA-DQA1b
30247	2	20	30	2	1	1301	0404	0103	...
980361	2	20	30	2	1	1501	0301	0102	...
980554	2	20	30	1	2	0101	1403	0101	...
980866	2	20	60	.	1	1501	0802	0102	...
950437	2	20	60	1	1	1301	1402	0103	...
890225	2	20	30	1	1	0403	0801	0401	...
990720	2	20	30	2	1	1101	0301	0501	...

APPENDIX B VALIDATION PIPELINE ERROR MESSAGES

There is several error categories defined for the validation pipeline. These categories are defined by a numeric range as follows in Table B.1, "Error Message Categories". Errors messages for each category are specified in the Subsections below. The notation, '**__N__**', represents the position of the explicit value for a given error.

Table B.1, Error Message Categories

Error Category	Error Number
Allele Validator	100000
Allele Disambiguator	500000
Allele	600000
Lookup Tables Manager	900000
PyPop Runner	1000000
Tools	-3000
HLA File	-1008000
HLA File Converter	-1009000

B.1 ALLELE VALIDATOR ERRORS

The Allele Validator error messages are specified in Table B.2, "Allele Validation Error Messages". These errors are not immediately fatal since they are registered (written to the log-file and counted). Anyone of these error messages will cause validation to ultimately fail at the end of the validation processing step.

Table B.2, Allele Validation Error Messages

Message Number	Error Message
100005	Allele cell component has an even (2) number of digits cell comp = __1__
100006	Allele cell component is a gcode with odd-number of digits locus_name = __1__ cell comp = __2__ type = __3__
100007	Allele cell component is an NMDP code without two-digits locus_name = __1__ cell comp = __2__ type = __3__
100008	Allele cell component has an odd (1 or 3) number of digits cell comp = __1__
100009	Allele cell component is a serological component Cell comp = __1__
100010	First NMDP 4-digit code has different first 2-digits than cell cell = __1__ cell digits = __2__ Nmdp data = (__3__)
100011	Gcode is not in the recognized list of gcodes Cell = __1__
100012	Allele cell component has a locus prefix and two digits Cell comp = __1__
100013	Allele cell component not found in current alleles locus = __1__ Comp = __2__
100014	Allele cell component not found in current alleles, but its first __3__ digits are present for alleles in locus locus = __1__ comp = __2__
100015	Allele cell component has odd-number of digits and is not changed locus = __1__ comp = __2__
100017	Allele cell skipped, since main component is full name with an odd (1 or 3) number of digits cell comp = __1__
100018	Allele cell skipped, since serological alleles cannot have an allele suffix cell comp = __1__
100019	Allele cell skipped, since cell contains multiple serological alleles cell = __1__ main comp = __2__
100021	Allele cell skipped, since multi-allele cell contains a gcode cell = __1__ cell type = __2__ main comp = __3__
100022	Allele cell component has been deleted, but not replaced comp = __1__
100023	Multi-allele cell contains an NMDP code comp = __1__
100024	Multi-allele cell contains a gcode comp = __1__
100025	Allele cell skipped, since multi-allele cell contains an NMDP code cell = __1__ cell type = __2__ main comp = __3__

B.2 TOOLS ERRORS

Tools errors are always fatal. Table B.3, "Tools Error Messages" provides the specific error messages.

Table B.3, Tools Error Messages

Message Number	Error Message
-3001	Cannot add error messages and categories for class class = __1__ errMsg = __2__
-3002	Cannot Find __3__: name = __1__ lib = __2__ type = __3__ item = __4__
-3003	Missing property property file = __1__ property = __2__
-3004	Cannot set execution focus executionDir = __1__
-3005	Error executing tool msg = __1__ cmd = __2__
-3006	Unknown status status = __1__
-3007	Unable to open status file for writing status file = __1__
-3008	Unable to open status file for reading status file = __1__
-3009	Cannot evaluate string eval_status = __1__ eval_str = __2__
-3010	Datum is not set datum type = __1__
-3011	Datum is not constant datum type = __1__ old datum = __2__ new datum = __3__
-3012	Unknown tool name tool name = __1__
-3013	Unknown pipeline type pipeline type = __1__
-3101	MHC File Type unknown mhcFileType = __1__
-3102	MHC Reader Type unknown mhcReaderType = __1__
-3103	MHC Object Type unknown objectType = __1__
-3104	Unknown file type (not tab-separated, '.txt', nor Excel spreadsheet '.xls') file = __1__
-3105	Cannot open and write to file type file file type file = __1__
-3106	Cannot determine HLA file type for file file = __1__

B.3 HLA File Errors

HLA File errors are always fatal. Table B.4, "HLA Error Messages" provides the specific error messages

Table B.4, HLA File Error Messages

Message Number	Error Message
-1008001	HLA Locus Name is not defined for taxon locus_name = __1__ taxon_id = __2__
-1008002	File type incorrect or did not find locus names file type = __1__ file type checked = __2__ header val = __3__ header val checked = __4__
-1008003	Filename is not tab-separated (ie, suffix '.txt') source = __1__ file = __2__
-1008004	Error opening tab-separated file to write data source = __1__ file = __2__
-1008005	Unknown file type file type = __1__

B.4 Allele Errors

The Allele error messages are specified in Table B.5, "Allele Error Messages". These errors are not immediately fatal since they are registered (written to the log-file and counted). Anyone of these error messages will cause the either validateAlleles.pl or disambiguateAlleleNames.pl script to ultimately fail at the end of the processing step.

Table B.5, Allele Error Messages

Message Number	Error Message
600001	Validated file is not tab-separated (.txt) file = __1__
600002	Allele cell component does not conform expected syntax locus_name = __1__ cell comp = __2__
600003	Allele cell component does not have an expected type locus_name = __1__ cell comp = __2__
600004	Allele cell component does not conform to expected length locus_name = __1__ cell comp = __2__ Digit length = __3__
600005	Allele cell skipped, since cell has syntax errors cell = __1__ cell type = __2__

B.5 HLA File Converter Errors

The HLA File Converter error messages are specified in Table B.6, "HLA File Converter Error Messages". These errors are not immediately fatal since they are registered (written to the

log-file and counted). Anyone of these error messages will cause the preProcessAlleleNames.pl or runPyPop.pl script to ultimately fail at the end of the processing step.

Table B.6, HLA File Converter Error Messages

Message Number	Error Message
-1009001	The source hla reader not of the correct class type (file::Hla) source reader type = __1__
-1009002	Destination hla file type is neither HLA Typing Template nor PyPop dest_type = __1__
-1009003	Filename is not tab-separated (ie, suffix '.txt') source = __1__ file = __2__
-1009004	Error opening tab-separated file to write data source = __1__ file = __2__
-1009005	Column Pair Does not have the same locus col_1 = __1__ locus_1 = __2__ col_2 = __3__ locus_2 = __4__

B.6 LOOKUP TABLE MANAGER ERRORS

The Lookup Table Manager errors are specified in Table B.7, "Lookup Table Manager Error Messages". These errors are not immediately fatal since they are registered (written to the log-file and counted). Anyone of these error messages will cause the prevalidateAlleleNames.pl or disambiguateAlleleNames.pl script to ultimately fail at the end of the processing step.

Table B.7, Lookup Table Manager Error Messages

Message Number	Error Message
900001	Cannot instantiate lookup table object eval_status = __1__ eval_str = __2__

B.7 ALLELE DISAMBIGUATOR ERRORS

The Allele Disambiguator error messages are specified in Table B.8, "Allele Disambiguation Error Messages". These errors are not immediately fatal since they are registered (written to the log-file and counted). Anyone of these error messages will cause disambiguation to ultimately fail at the end of the disambiguation processing step.

Table B.8, Allele Disambiguation Error Messages

Message Number	Error Message
500001	gcode is not in the recognized list of gcodes gcode = __1__
500002	NMDP code is not in the recognized list of NMDP codes nmdp code = __1__
500003	Cell containing multiple alleles contains serological code sero code = __1__
500004	Cell containing multiple alleles contains an NMDP code nmdp code = __1__

Message Number	Error Message
500005	Cell containing multiple alleles contains a gcode gcode = __1__
500006	Cell containing multiple alleles contains more than one gcode and/or cwd allele alleles = __1__ cell returned = __2__ gcodes = __3__ cwd alleles = __4__ rare alleles = __5__

B.8 PyPop Runner Errors

The PyPop Runner error messages are specified in Table B.9, "PyPop Runner Error Messages". These errors are not immediately fatal since they are registered (written to the log-file and counted). Anyone of these error messages will cause the execution of the PyPop tool to ultimately fail to execute.

Table B.9, PyPop Runner Error Messages

Message Number	Error Message
1000001	Error opening pypop config file pypop file = __1__ config file = __2__
1000002	PyPop config category missing category = __1__
1000003	PyPop property is not defined correctly the properties property = __1__
1000004	pypopCategories is not a Perl referenced hash property = __1__ value = __2__
1000005	Header row has not been set